

Joseph Olive
Caitlin Christianson
John McCary *Editors*

Handbook of Natural Language Processing and Machine Translation

DARPA Global Autonomous
Language Exploitation

 Springer

Handbook of Natural Language Processing and Machine Translation

Joseph Olive · Caitlin Christianson · John McCary
Editors

Handbook of Natural Language Processing and Machine Translation

DARPA Global Autonomous Language Exploitation

 Springer

Editors

Joseph Olive
Defense Advanced Research Projects Agency
IPTO
N Fairfax Drive 3701
Arlington, VA 22203, USA
joseph.olive@darpa.mil

Caitlin Christianson
Defense Advanced Research Projects Agency
Reston Virginia, USA
caitlin.christianson.ctr@darpa.mil

John McCary
Defense Advanced Research Projects Agency
Bethesda Maryland, USA
john.mccary@gmail.com

ISBN 978-1-4419-7712-0 e-ISBN 978-1-4419-7713-7
DOI 10.1007/978-1-4419-7713-7
Springer New York Dordrecht Heidelberg London

Library of Congress Control Number: 2011920954

© Springer Science+Business Media, LLC 2011

All rights reserved. This work may not be translated or copied in whole or in part without the written permission of the publisher (Springer Science+Business Media, LLC, 233 Spring Street, New York, NY 10013, USA), except for brief excerpts in connection with reviews or scholarly analysis. Use in connection with any form of information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed is forbidden.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Acknowledgements

First, I would like to thank the community for all of its hard work in making GALE a success. I would like to thank the technical assistants Caitlin Christianson and John McCary for all of their help with the program and this book. Special thanks to Paul Dietrich for making GALE run smoothly.

We would like to thank the current DARPA management for its continued support throughout the program, especially DARPA Director Regina Dugan, Deputy DARPA Director Ken Gabriel, Information Processing Techniques Office (IPTO) Director Daniel Kaufman, and IPTO Deputy Director Mark Luetgen. We would also like to thank previous IPTO Directors and Deputy Directors Ron Brachman, Charles Holland, Barbara Yoon, Charles Holland, and Charles Morefield for their help in launching the program, continued support, and encouragement to write this book.

Special thanks to former DARPA Director Anthony Tether for having the vision and the faith in us to fund the program so that it had a chance to succeed.

Finally, I would like to thank my wife Virginia Kayhart Olive. Without her great personal sacrifice this project would not have been possible.

Introduction

Authors: Joseph Olive, Caitlin Christianson, and John McCary

“When I use a word, it means just what I choose it to mean – neither more nor less.”
Humpty Dumpty in Lewis Carroll’s *Through the Looking Glass, or Alice in Wonderland*

The meaning of what Lewis Carroll’s Humpty Dumpty says is abundantly clear to him, but to others, it is virtually incomprehensible. Because the rules of his language are entirely of his own devising, he is the only one who knows what he means. Translation of one language into another poses a similar problem; knowing the usual meaning of a word is not enough. To translate, it is necessary to convey the meaning of the entire message, not just transfer words from one language to another. Because people can perform this task so adeptly, it is easy to underestimate the challenge it poses to computers. Although computational capabilities of machines exceed those of humans in many ways, even the most advanced of today’s computers cannot match the language ability that humans acquire naturally. To translate and extract information conveyed through language, humans take advantage of a variety of cognitive abilities that no computer can currently emulate. The Defense Advanced Research Projects Agency (DARPA), however, specializes in tackling just such challenging problems. DARPA researchers have attacked the problem of machine translation as part of the Global Autonomous Language Exploitation (GALE) Program.

Like other programs at DARPA, GALE was initiated to fill a need for the Defense Department. In the case of GALE, this need is to close the language gap – to make relevant information, regardless of source language, accessible to English-speaking personnel. The program’s goal is to create technology to automatically translate, analyze, and distill the information that forms the overwhelming tidal wave of foreign language material facing Defense Department personnel. The value of GALE technology is to be creation of the ability not only to translate, but also to identify relevant information and sort it from what is not, and to present a final product in a form both understandable and useable.

While there is no existing parallel for such a capability, there have certainly been fictionalized precedents to the idea behind GALE – the universal translator, capable of translating between English and thousands of other languages, in the form of a compact silver device worn on the chest of every *Star Trek* crew member; the HAL 9000 computer, capable of reasoning and defending itself in *2001*, Stanley Kubrick’s classic

futuristic film; and *Star Wars*' robot C3PO, who speaks nearly every known language in the universe.

For the purpose of research and development of automated translation and human language processing capability in the GALE Program, language has been classified into two input modes— speech and electronic text. A third important language input mode, hardcopy document text, is the subject of another DARPA program. While research related to each of GALE's two input modes focuses on producing a correct translation and extracting information, each mode of input presents singular problems that require different research paths. Unlike text, which generally has orthographic separation, speech signals are continuous, lacking word and phrase boundary markers. In speech, even sentence boundaries are difficult to determine. In addition, the confusability of many phonemes adds to the uncertainty. Because these difficulties are alleviated when a computer carries out transcription using a more explicit orthographic system than those used by human writers, machine translation researchers previously attempted to use computers to transcribe speech into text and then translate that text.

In the GALE Program, researchers have begun to combine the processes of transcription and translation, enabling information about possible translations to assist transcription, and information about possible transcriptions and transcription ambiguities to assist translation. An important benefit of using this method is that interactivity between transcription and translation reduces instances of propagation of errors made earlier in the process by providing opportunities for correction. This interactive technique has yielded significant improvements in accuracy for both transcription and translation of speech. In this way, GALE researchers have achieved revolutionary progress by consistently and effectively blending previously distinct speech and text technologies.

Text input can also be problematic. In many languages, word boundaries are clear in writing, but due to the lack of orthographic representation of other language elements, such as prosody, it can still be difficult to know a writer's intent. Scripts without word boundaries introduce additional uncertainty in reading. Chinese writing, for example, does not indicate word boundaries orthographically, a characteristic that can create ambiguity. Semitic language scripts do indicate word boundaries, but often do not include explicit vowel marking, thus creating ambiguities, since it can be uncertain which vowels were intended. GALE researchers have undertaken extraordinary efforts to address these and other obstacles in machine translation of text.

One of the greatest challenges in planning an approach for GALE was defining precisely what tasks GALE's natural language processing machines would be achieving. Was it the ability to translate any language into English? Or was there an even higher goal of retrieving what was relevant from the input? Would achieving such a goal mean that GALE researchers would have to create technology that could extract relevant information from translated material and operate on foreign language material directly? Would GALE machines be able to perform all of these tasks well enough to enable assessment and analysis of the volume of information now available to anyone connected to the Internet or satellite television?

These questions have resulted in many challenges to GALE and refinements of the program's fundamental aspects.

A Partial History of Human Language Technology Research at DARPA

Author: Allen Sears

During the past four decades, DARPA has sponsored a wide variety of research on human language technology – efforts that turned out to be stepping stones to GALE.

DARPA entered the speech recognition field in 1971 with the launch of the five-year Speech Understanding Research (SUR) Program. Although its immediate impact was limited, SUR included pioneering work with hidden Markov models, which lie at the heart of all modern speech-to-text systems.

DARPA speech and text processing research proceeded at a relatively low level from the late 1970s through the early 1980s, then accelerated in the second half of the 1980s.

On the speech side, the Spoken Language Program worked on automatic transcription of grammatically constrained read speech with a 1000-word vocabulary, advancing from speaker-dependent to speaker-independent transcription. In the early 1990s, the program moved on to include read speech from *Wall Street Journal* sentences, progressing from a 5000-word vocabulary, through a 20,000-word vocabulary, to an unlimited vocabulary. A companion program, WHISPER, made an initial foray into automatic transcription of conversational telephone speech.

On the text side, the Written Language Program began working on technology to pull facts out of short, semantically-constrained military reports. In the early and mid 1990s, the TIPSTER program aggressively tackled the twin challenges of detecting relevant documents and extracting information needed to fill templates, using naturally-occurring English and Japanese documents as source data. And in the early 1990s, a modest DARPA machine translation initiative explored competing approaches for translating unconstrained foreign language text, laying important groundwork for future advances.

In the mid and late 1990s, DARPA's Text, Radio, Video, Speech (TRVS) Program worked on transcribing and analyzing broadcast news, emphasizing English, but including some preliminary work on Arabic and Chinese. In the late 1990s, Topic Detection and Tracking (TDT) attacked the problem of finding and following events discussed in news reports. In the early 2000's, Automatic Content Extraction (ACE) made a fresh assault on the challenge of discovering and characterizing entities, relations and events described in newswire plus automatically transcribed broadcast news.

Two major programs launched in the early 2000's were particularly significant —

Effective, Affordable, Reusable Speech-to-Text (EARS) attacked challenges posed by broadcast news and telephone conversations in English, Chinese, and Arabic. In addition to improving the speed and accuracy of transcription, EARS worked on automatic metadata extraction to make transcripts more readable by adding structure and removing disfluencies.

Translingual Information Detection, Extraction, and Summarization (TIDES) worked towards enabling English speakers to find and interpret required information quickly and effectively regardless of language or medium. TIDES dealt with input from a variety of sources including newswires and automatically transcribed broadcast news in

English, Arabic, and Chinese. It also included “surprise language experiments” that showed how well and how quickly the technology could be ported to other languages.

To meet the challenges posed by real world data, DARPA researchers developed increasingly sophisticated algorithms, moving away from symbolic approaches that relied on hand-coded rules and towards statistical approaches that learned from large quantities of sample data and were substantially language-independent. The shift from symbolic to statistical approaches occurred over a number of years. It happened first in the speech community, where a 1987 evaluation of automatic transcription algorithms put to rest the notion that good speech-to-text systems could be built from hand-coded rules. The text processing community followed suit, learning a great deal from the speech community.

Building on the advances in automatic transcription and translation achieved by EARS and TIDES, DARPA produced two multilingual news monitoring systems (eTAP and TALES) able to convert Arabic and Chinese broadcasts to English good enough for English speakers to find relevant material. Deployed to military customers (CENTCOM and PACOM), these systems were productively employed from 2004 onwards.

Three and a half decades of progress had begun to produce useful technology and provided a strong foundation. It was time for a grander and more ambitious program, GALE.

The GALE Program

Authors: Joseph Olive and Caitlin Christianson

Planning GALE

The most fundamental difference between GALE and its predecessor programs has been its holistic integration of previously separate or sequential processes. In earlier language programs, each of the individual component processes was carried out individually: speech recognition, transcription, translation, information retrieval, content extraction, and content presentation. GALE involves use of a distinctly new approach, one by which researchers have sought to create systems able to execute these processes simultaneously. Under this rubric, speech transcription algorithms aid translation and vice versa. In addition, the processes of information retrieval, content extraction, and content presentation have been joined into an activity referred to under GALE as distillation, which has also been included in the interactive assistance framework of transcription and translation. As is further detailed in the chapters that follow, this combination of previously distinct processes has resulted in substantial technological breakthroughs.

The GALE program focuses primarily on transcription, translation, and distillation of information in two languages: Mandarin Chinese and Arabic. These two languages have been chosen because of the high degree of difficulty posed by translation between each of them and English, their relative linguistic distance from each other and from English, the relatively high availability of data in both, and their immediate relevance to current national security applications.

The Origin of the GALE Program

Around 2004, DARPA Director Anthony Tether asked two important questions regarding human language technology programs at DARPA: who the end users were, and at what level of accuracy the technologies would become useful. In part, these questions arose from the reduction in word error rate (WER) that resulted from the EARS Program, but it was not clear toward what goal this reduction was aimed. For dictation, WER above 10 percent is not acceptable, but for a dialogue system, a much higher rate can be tolerated as long as the system can enable a user to complete a specific task successfully.

Initial studies were conducted to determine answers to these questions, but the results were not satisfactory, mainly due to the fact that the stimuli used did not represent a fine enough grain to determine what level of accuracy was sufficient. For example, translation quality testing was performed on only two intermediate levels of translation quality between machine-generated translation quality and human-generated translation quality. The stimuli consisted of translation in which either one or two of every three sentences was machine generated and the remaining sentences were human generated.

To get better answers to the questions of language technology applications and the quality of output required for these applications, it was necessary to design a new study. With the help of retired Air Force Colonel Jose Negron, DARPA contacted Colonel Rafael Sanchez-Carrasquillo, head of a group of language analysts at the Defense Intelligence Agency to ask him if, based on his experience, he could answer DARPA's question about the level of accuracy at which a translation would become useful for various analysis tasks. Colonel Sanchez-Carrasquillo agreed to provide assistance in carrying out a study to determine the answer, so the next task was to create a set of translations at accuracy levels between those of baseline machine translation and human translation with quality gradations small enough to enable determination of the level of accuracy necessary to enable performance of various tasks. With assistance from Kevin Knight and Salim Roukos, two representatives of the machine translation community, the following process was created. First, machine translation was corrected by a human so that the edited translation reflected the meaning of the original document. Dividing the number of edits by the number of words in the translated passage resulted in an accuracy value of 55 percent for both Arabic and Spanish. Randomly-selected errors were then removed 5 percent at a time to create ten translations, ranging from 55 percent to 100 percent accurate relative to the human-translated standard. These translations of varying accuracy were then presented to analysts from the intelligence community and Defense Department, who were asked to determine what quality of translation would be appropriate for gisting, triage, editing, or use without any alteration. While there was no overwhelming consensus among the analysts as to the level of accuracy at which a translation was felt to be useful, there was a sense that an accuracy level between 75 percent and 80 percent was required to gain a basic understanding of the meaning of a passage. For a translation to be truly useful, however, the intelligence analysts chose the 90 percent mark. They stated that with translation at or above a 90 percent accuracy level, they would choose to work with an existing translation, making edits improve its quality, rather than starting from scratch. The 90th percentile was therefore determined to

be the standard for a translation to be deemed edit-worthy, and would become an important programmatic target for GALE. It is important to note, however, that translations created by human professionals often do not meet this mark without multiple stages of revision.

Evaluation under GALE

Prior to GALE, many translation programs relied on the BLEU metric (See Section 5.2.2.2), an automatic system for evaluating translation quality by counting word or word group matches between a machine-generated translation and multiple human-created translations. The fact that this means of evaluation was automatic rather than manual allowed algorithm developers to conduct numerous experiments in a relatively short period of time, which enabled great progress in machine translation. Despite comparing each machine translation to multiple human translations, however, there was no guarantee that scores generated by BLEU would correlate with preservation of the meaning of the source-language document in the translation.

Instead of using an automatic metric, it was determined that GALE machine translation systems would be evaluated on the basis of whether their output accurately conveyed the correct meaning of the source language in English, allowing for different but equivalent word choice and word order. The evaluation standard used in GALE has become known as human translation error rate (HTER) and is based on edit distance. For GALE purposes, edit distance is determined as the number of edits an editor is required to make to a machine-generated translation for it to accurately reflect the meaning of a corresponding highly perfected human translation created through multiple translators and multiple levels of revision, i.e. a “gold standard” translation. The process for creating gold standard translations was developed in cooperation with National Virtual Translation Center Technology Director Kathleen Egan and Stephanie Strassel of the Linguistic Data Consortium.

The ultimate goal of the GALE Program has been set as achievement of 95 percent accuracy in translation of Arabic and Chinese newswire text and broadcast news speech into English text. Also, in response to requirements gathered from potential users of GALE technologies, additional genres have been added, such as talk shows, newsgroups, and weblogs. For these less formal genres, which pose a higher degree of difficulty to both human translators and machine translation systems than do the more formal genres, target accuracy has been set at 85 percent. At the end of GALE’s first year, the goals for that year had been achieved, but DARPA’s director revised the targets to make the task more difficult by specifying that future target accuracy levels would not be averages, but minimums that had to be met for a certain percentage of documents in each test set. The resulting goals follow a gradually increasing scale, specifying that translation accuracy of 95 percent must be achieved for 95 percent of documents in the relevant test set for newswire, with slightly lower targets for the other, more difficult genres (See section 5.4.4.8.1 for all GALE targets).

To accomplish these ambitious targets for speech input, it was proposed that there be combination of translation and transcription technologies, which had previously been developed separately, so that errors in transcription would not necessarily be

irrecoverable in translation. For all translation, it was also proposed that there be incorporation of a variety of algorithms, including algorithms relating to morphology, syntax, semantics, and topic-dependent language models. In GALE, researchers have also stopped relying on the unsustainable process of developing extremely large parallel corpora, for which matching sets of heavily annotated transcripts in both English and a corresponding foreign language must be obtained or created. This step has been taken under GALE due to the fact that although machine translation accuracy does increase when systems are trained with increasingly large amounts of parallel data, each degree of accuracy improvement requires an exponentially larger amount of data.

GALE Data

The approach adopted in GALE represents a shift from the concept of employment of increasingly large amounts of data to the use of smaller amounts of richer data. As there was already a huge amount of newswire and broadcast news parallel data at the beginning of the program, some new data has been added to these genres, but the greatest portion of the funds for data collection have been invested in targeted collection of data to address particular areas of difficulty, as well as annotation, such as treebanks, propbanks, careful text alignment, etc. In addition, corpora used in GALE have been augmented by addition of collections from other language programs such as the Text Retrieval Evaluation Conference.

GALE Distillation

In view of the ever-increasing amount of information confronting those responsible for maintaining the security of the United States, the decision was made that GALE would not just address the most immediate challenges in human language processing of transcription and translation, but also take human language technology research a step further, towards determining how to use all the new information made accessible by automatic transcription and translation.

To answer this challenge, GALE has included, as a second and integral step in the language processing paradigm, the assessment, analysis, and presentation of translation results in an easily readable and coherent format. This process consists of a combination of information retrieval, content extraction, and content presentation, which have been collectively termed distillation. Distillation is a concept entirely new in GALE, in which relevant information is extracted from foreign language and English input and concisely presented to the user in English. GALE distillation is not just a key-word search, and does not involve summarization. Instead, it consists of utilization of language analysis to identify information relevant to a user's query, with the aim of extracting all available relevant information without redundancy and presenting it to the user in a functional form.

Because warfighters and analysts often face time pressure, they do not have the luxury of wading through many documents that present similar information. Therefore, GALE distillation entails placing particular importance on targeted searches and elimination of redundant results. GALE distillation includes a goal of combining

redundant results and presenting users with a single, distilled version of what is important, accompanied by multiple citations. Depending on the intended task of a user – enabling military operations, conducting intelligence analysis, assisting policy formulation, monitoring foreign perceptions – GALE systems are required to provide a customized version of a given set of data, saving users hours, if not days or weeks, and allowing a user’s valuable energy and insight to be focused on only the most important information. Through translation in combination with distillation, GALE systems are intended to increase the number of sources of information available by translating previously inaccessible foreign language data and the efficiency of system users in employing this newly-available data to conduct whatever task is required.

Like the GALE translation goals, GALE distillation performance targets have also been set very high, with a final target of 95 percent recall and 90 percent precision. To give system performance a meaningful standard of measurement, computer distillation performance is compared to that of a human given the same requirements.

The research detailed in this book shows a snapshot of the results of the first three years of groundbreaking progress under the GALE Program. As of the writing of this book, two years of GALE research remain.

Contents

1	Data Acquisition and Linguistic Resources	1
1.1	Introduction.....	1
1.2	Data Collection, Distribution, and Management.....	2
1.3	Human Annotation.....	14
1.4	Automatic Annotation.....	64
2	Machine Translation from Text	133
2.1	Introduction.....	133
2.2	Segmentation, Tokenization and Preprocessing.....	135
2.3	Word Alignment	164
2.4	Translation Models	183
2.5	Language Modeling for SMT	252
2.6	Search and Complexity	271
2.7	Adaptation and Data Selection.....	297
2.8	System Combination.....	324
3	Machine Translation from Speech	399
3.1	Introduction.....	399
3.2	Front End Features.....	401
3.3	Improved Speech Acoustic Models	428
3.4	Language Models.....	460
3.5	Language-Specific Models and Systems: Mandarin.....	485
3.6	Language-Specific Models and Systems: Arabic.....	520
3.7	Integration of Speech Recognition and Translation.....	569
4	Distillation	617
4.1	Introduction.....	617
4.2	Template-Based Query Development.....	618
4.3	Architecture and Implementation of a Distillation System.....	623
4.4	Enabling Technology Breakthroughs to Improve Distillation Capabilities	636
4.5	Distillation in an Integrated GALE system.....	690
4.6	Evaluating Distillation Technology	716
5	Machine Translation Evaluation and Optimization	745
5.1	Introduction.....	745
5.2	Automatic and Semi-Automatic Measures	758
5.3	Tasks and Human-in-the-Loop Measures	768

5.4 GALE Machine Translation Metrology: Definition, Implementation, and Calculation	783
5.5 Use of Evaluation for Optimization	812
5.6 Searching for Better Automatic MT Metrics	818
6 Operational Engines	845
6.1 Introduction	845
6.2 Implementation of Operational Engines	846
6.3 Evaluation of Operational Engines	905
Concluding Remarks	933

Contributors

- Abhaya Agarwal** Carnegie Mellon University, Pittsburgh, PA, USA
- Jaewook Ahn** University of Pittsburgh, Pittsburgh, PA, USA
- James Allan** University of Massachusetts Amherst, Amherst, MA, USA
- Abhishek Arun** University of Edinburgh, Edinburgh, UK
- Sabine Atwell** Defense Language Institute, Monterey, CA, USA
- Necip Fazil Ayan** SRI International, Menlo Park, CA, USA
- Olga Babko-Malaya** BAE Systems, Burlington, MA, USA
- Robert Belvin** HRL Laboratories, Malibu, CA, USA
- Oliver Bender** RWTH Aachen University, Aachen, Germany
- Ann Bies** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Daniel M. Bikel** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Maximilian Bisani** RWTH Aachen University, Aachen, Germany
- Matthias Blume** Fair Isaac Corporation, San Diego, CA, USA
- Roger Bock** BBN Technologies, Cambridge, MA, USA
- Elizabeth Boschee** BBN Technologies, Cambridge, MA, USA
- Sebastián Bronsart** National Institute of Standards and Technology, Gaithersburg, MD, USA
- Peter Brusilovsky** University of Pittsburgh, Pittsburgh, PA, USA
- William Byrne** Cambridge University, Cambridge, UK
- Marine Carpuat** Columbia University, New York, NY, USA
- Christopher Caruso** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Vittorio Castelli** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Ozgur Cetin** International Computer Science Institute, Berkeley, CA, USA
- Achraf Chalabi** Sakhr Software, Vienna, VA, USA
- Pi-Chuan Chang** Stanford University, Stanford, CA, USA
- Upendra V. Chaudhari** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Caitlin Christianson** Defense Advanced Research Projects Agency, Arlington, VA, USA

- Stephen M. Chu** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Christopher Cieri** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Sean Colbath** BBN Technologies, Cambridge, MA, USA
- Steve DeNeefe** Information Sciences Institute, University of Southern California, Los Angeles, CA, USA
- Michael Denkowski** Carnegie Mellon University, Pittsburgh, PA, USA
- Thomas Deselaers** RWTH Aachen University, Aachen, Germany
- Mona T. Diab** Columbia University, New York, NY, USA
- Frank Diehl** Cambridge University, Cambridge, UK
- Dan Ding** Defense Language Institute, Monterey, CA, USA
- Denise DiPersio** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Bonnie Dorr** University of Maryland, College Park, MD, USA
- Loïc Dugast** SYSTRAN Software, Inc., San Diego, CA, USA
- Chris Dyer** University of Maryland, College Park, MD, USA
- Abdessamad Echihabi** Language Weaver, Los Angeles, CA, USA
- Kathleen Egan** Department of Defense, USA
- Jason Eisner** Johns Hopkins University, Baltimore, MD, USA
- Ahmad Emami** Johns Hopkins University, Baltimore, MD, USA
- Edward A. Epstein** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Reem Faraj** Columbia University, New York, NY, USA
- Arlo Faria** International Computer Science Institute, Berkeley, CA, USA
- Benoit Favre** International Computer Science Institute, Berkeley, CA, USA
- David Ferrucci** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Radu Hans Florian** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- George Foster** National Research Institute of Canada, Saskatoon, SK, Canada
- Connie Fournelle** BAE Systems, Burlington, MA, USA
- Petr Fousek** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Marjorie Freedman** BBN Technologies, Cambridge, MA, USA
- Dayne Freitag** Fair Isaac Corporation, San Diego, CA, USA

- Lauren Friedman** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Mark Fuhs** Carnegie Mellon University, Pittsburgh, PA, USA
- Pascale Fung** The Hong Kong University of Science and Technology, Hong Kong, China
- Mark J.F. Gales** Cambridge University, Cambridge, UK
- Michel Galley** Stanford University, Stanford, CA, USA
- Jianfeng Gao** Microsoft Corporation, Redmond, WA, USA
- Qin Gao** Carnegie Mellon University, Pittsburgh, PA, USA
- Jean-Luc Gauvain** The Computer Sciences Laboratory for Mechanics and Engineering Sciences (LIMSI), Orsay Cedex, France
- Niyu Ge** IBM T.J. Watson Research Center, Yorktown Heights, NY, USA
- Daniel Gillick** International Computer Science Institute, Berkeley, CA, USA
- Adrià de Gispert** Cambridge University, Cambridge, UK
- Meghan Lammie Glenn** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Christian Gollan** RWTH Aachen University, Aachen, Germany
- Martin Graciarena** SRI International, Menlo Park, CA, USA
- Jonathan Grady** University of Pittsburgh, Pittsburgh, PA, USA
- John Graettinger** BBN Technologies, Cambridge, MA, USA
- Stephen Grimes** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Ralph Grishman** New York University, New York, NY, USA
- Francisco Guzman** Carnegie Mellon University, Pittsburgh, PA, USA
- Nizar Habash** Columbia University, New York, NY, USA
- Dilek Hakkani-Tür** International Computer Science Institute, Berkeley, CA, USA
- Greg Hanneman** Carnegie Mellon University, Pittsburgh, PA, USA
- Mary Harper** University of Maryland, College Park, MD, USA
- Saša Hasan** RWTH Aachen University, Aachen, Germany
- Daqing He** University of Pittsburgh, Pittsburgh, PA, USA
- Kenneth Heafield** Carnegie Mellon University, Pittsburgh, PA, USA
- Georg Heigold** RWTH Aachen University, Aachen, Germany

- Hynek Hermansky** International Computer Science Institute, Berkeley, CA, USA
- Martha Herzog** Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, MA, USA
- Almut Silja Hildebrand** Carnegie Mellon University, Pittsburgh, PA, USA
- Dustin Hillard** University of Washington, Seattle, WA, USA
- Julia Hirschberg** Columbia University, New York, NY, USA
- Hieu Hoang** University of Edinburgh, Edinburgh, UK
- Bjorn Hoffmeister** RWTH Aachen University, Aachen, Germany
- Jon Holbrook** Aptima, Inc., Woburn, MA, USA
- Eduard Hovy** Information Sciences Institute, University of Southern California, Los Angeles, CA, USA
- Roger Hsiao** Carnegie Mellon University, Pittsburgh, PA, USA
- Fei Huang** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Zhongqiang Huang** University of Maryland, College Park, MD, USA
- Dan Hunter** BAE Systems, Burlington, MA, USA
- Mei-Yuh Hwang** University of Washington, Seattle, WA, USA
- Hussny Ibrahim** Defense Language Institute, Monterey, CA, USA
- Abraham Ittycheriah** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Heng Ji** New York University, New York, NY, USA
- Qin Jin** Carnegie Mellon University, Pittsburgh, PA, USA
- Doug Jones** Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, MA, USA
- Jeremy Kahn** University of Washington, Seattle, WA, USA
- Damianos Karakos** Johns Hopkins University, Baltimore, MD, USA
- Shahram Khadivi** RWTH Aachen University, Aachen, Germany
- Sanjeev Khudanpur** Johns Hopkins University, Baltimore, MD, USA
- Daniel Kiecza** BBN Technologies, Cambridge, MA, USA
- Brian Kingsbury** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Katrin Kirchhoff** University of Washington, Seattle, WA, USA
- Judith L. Klavans** University of Maryland, College Park, MD, USA
- Kevin Knight** Information Sciences Institute, University of Southern California, Los Angeles, CA, USA

- Philipp Koehn** University of Edinburgh, Edinburgh, UK
- Gary Krug** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Roland Kuhn** National Research Institute of Canada, Saskatoon, SK, Canada
- Seth Kulick** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Hong-Kwang Kuo** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Lori Lamel** The Computer Sciences Laboratory for Mechanics and Engineering Sciences (LIMSI), Orsay Cedex, France
- Ian Lane** Carnegie Mellon University, Pittsburgh, PA, USA
- Alon Lavie** Carnegie Mellon University, Pittsburgh, PA, USA
- Audrey Le** National Institute of Standards and Technology, Gaithersburg, MD, USA
- Haejoong Lee** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Xin Lei** SRI International, Menlo Park, CA, USA
- Gregor Leusch** RWTH Aachen University, Aachen, Germany
- Michael Levit** International Computer Science Institute, Berkeley, CA, USA
- Burn L. Lewis** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Xuansong Li** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Zhifei Li** Johns Hopkins University, Baltimore, MD, USA
- Martha Lillie** BBN Technologies, Cambridge, MA, USA
- Xunying Andrew Liu** Cambridge University, Cambridge, UK
- Chi-kiu Lo** The Hong Kong University of Science and Technology, Hong Kong, China
- Tomasz Loboda** University of Pittsburgh, Pittsburgh, PA, USA
- Jun Luo** University of Maryland, College Park, MD, USA
- Xiaoqiang Luo** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Jeff Ma** BBN Technologies, Cambridge, MA, USA
- Weiyun Ma** Columbia University, New York, NY, USA
- Xiaoyi Ma** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Mohamed Maamouri** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Jessica MacBride** BBN Technologies, Cambridge, MA, USA

- Nitin Madnani** University of Maryland, College Park, MD, USA
- Carl Madson** SRI International, Menlo Park, CA, USA
- Kazuaki Maeda** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- John Makhoul** BBN Technologies, Cambridge, MA, USA
- Arindam Mandal** SRI International, Menlo Park, CA, USA
- Lidia Mangu** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Christopher D. Manning** Stanford University, Stanford, CA, USA
- Daniel Marcu** Language Weaver, Los Angeles, CA, USA
- Mitchell Marcus** University of Pennsylvania, Philadelphia, PA, USA
- Marie-Catherine de Marneffe** Stanford University, Stanford, CA, USA
- Spyros Matsoukas** BBN Technologies, Cambridge, MA, USA
- Evgeny Matusov** RWTH Aachen University, Aachen, Germany
- Arne Mauser** RWTH Aachen University, Aachen, Germany
- Andrea Mazzucchi** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Abdelkhalek Messaoudi** The Computer Sciences Laboratory for Mechanics and Engineering Sciences (LIMSI), Orsay Cedex, France
- Scott McCarley** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- John McCary** Defense Advanced Research Projects Agency, Arlington, VA, USA
- Kathleen McKeown** Columbia University, New York, NY, USA
- Calandra Tate Moore** University of Maryland, College Park, MD, USA
- Nelson Morgan** International Computer Science Institute, Berkeley, CA, USA
- Smaranda Muresan** Rutgers University, New Brunswick, NJ, USA
- Hazem Nader** Sakhr Software, Vienna, VA, USA
- Udhyakumar Nallasamy** Carnegie Mellon University, Pittsburgh, PA, USA
- Prem Natarajan** BBN Technologies, Cambridge, MA, USA
- Hermann Ney** RWTH Aachen University, Aachen, Germany
- Tim Ng** BBN Technologies, Cambridge, MA, USA
- Kham Nguyen** BBN Technologies, Cambridge, MA, USA
- Long Nguyen** BBN Technologies, Cambridge, MA, USA
- Jan Niehues** Institute for Theoretical Computer Science, Zürich, Switzerland

- Mohamed Noamany** Carnegie Mellon University, Pittsburgh, PA, USA
- Eric Nyberg** Carnegie Mellon University, Pittsburgh, PA, USA
- Douglas W. Oard** University of Maryland, College Park, MD, USA
- Joseph Olive** Defense Advanced Research Projects Agency, Arlington, VA, USA
- Mari Ostendorf** University of Washington, Seattle, WA, USA
- Sebastian Pado** Stanford University, Stanford, CA, USA
- Martha Palmer** University of Colorado at Boulder, Boulder, CO, USA
- Kishore Papineni** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Junho Park** Cambridge University, Cambridge, UK
- Robert Parker** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Alok Parlikar** Carnegie Mellon University, Pittsburgh, PA, USA
- Kristen Parton** Columbia University, New York, NY, USA
- Matthias Paulik** Carnegie Mellon University, Pittsburgh, PA, USA
- Jason Pelecanos** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- John F. Pitrelli** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Christian Plahl** RWTH Aachen University, Aachen, Germany
- Daniel Povey** Microsoft Corporation, Redmond, WA, USA
- Sameer Pradhan** BBN Technologies, Cambridge, MA, USA
- Mark Przybocki** National Institute of Standards and Technology, Gaithersburg, MD, USA
- Leiming Qian** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Yong Qin** IBM China Research Lab, Beijing, China
- Jerry Quinn** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Anna N. Rafferty** University of California, Berkeley, CA, USA
- Owen Rambow** Columbia University, New York, NY, USA
- Lance Ramshaw** BBN Technologies, Cambridge, MA, USA
- Suman Ravuri** Columbia University, New York, NY, USA
- Philip Resnik** University of Maryland, College Park, MD, USA
- Eric Riebling** Carnegie Mellon University, Pittsburgh, PA, USA
- Brian Roark** Oregon Health & Sciences University, Portland, OR, USA

- Monica Rogati** Carnegie Mellon University, Pittsburgh, PA, USA
- Antti-Veikko I. Rosti** BBN Technologies, Cambridge, MA, USA
- Ryan M. Roth** Columbia University, New York, NY, USA
- David Rybach** RWTH Aachen University, Aachen, Germany
- Salim Roukos** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Fatiha Sadat** University of Quebec Montreal, Montreal, QC, Canada
- Rami Safadi** Sakhr Software, Vienna, VA, USA
- Guruprasad Saikumar** BBN Technologies, Cambridge, MA, USA
- William Salter** Aptima, Inc., Woburn, MA, USA
- Gregory Sanders** National Institute of Standards and Technology, Gaithersburg, MD, USA
- George Saon** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Ralf Schlüter** RWTH Aachen University, Aachen, Germany
- Tanja Schultz** Carnegie Mellon University, Pittsburgh, PA, USA
- Richard Schwartz** BBN Technologies, Cambridge, MA, USA
- Holger Schwenk** University of Le Mans, Le Mans, France
- Allen Sears** Corporation for National Research Initiatives, Reston, VA, USA
- Jean Senellart** SYSTRAN Software, Inc., San Diego, CA, USA
- Libin Shen** BBN Technologies, Cambridge, MA, USA
- Wade Shen** Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, MA, USA
- Qin Shi** IBM China Research Lab, Beijing, China
- Heather Simpson** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Adish Singla** International Computer Science Institute, Berkeley, CA, USA
- Jason Smith** Johns Hopkins University, Baltimore, MD, USA
- Matthew Snover** University of Maryland, College Park, MD, USA
- Dagobert Soergel** University of Maryland, College Park, MD, USA
- Hagen Soltau** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Zhiyi Song** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Jeffrey Sorensen** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

Amit Srivastava BBN Technologies, Cambridge, MA, USA

William Staderman Defense Advanced Research Projects Agency, Arlington, VA, USA

Daniel Stein RWTH Aachen University, Aachen, Germany

Jens Stephan SYSTRAN Software, Inc., San Diego, CA, USA

Andreas Stolcke SRI International, Menlo Park, CA, USA

Stephanie Strassel Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA

David Svoboda Carnegie Mellon University, Pittsburgh, PA, USA

Yik-Cheung Tam Carnegie Mellon University, Pittsburgh, PA, USA

Christoph Tillmann IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

Marcus Tomalin Cambridge University, Cambridge, UK

Kristina Toutanova Microsoft Corporation, Redmond, WA, USA

Gokhan Tür SRI International, Menlo Park, CA, USA

Nicola Ueffing National Research Institute of Canada, Saskatoon, SK, Canada

Fabio Valente IDIAP Research Institute, Martigny, Switzerland

Dimitra Vergyri SRI International, Menlo Park, CA, USA

David Vilar RWTH Aachen University, Aachen, Germany

Paola Virga IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

Stephan Vogel Carnegie Mellon University, Pittsburgh, PA, USA

Clare Voss University of Maryland, College Park, MD, USA

Kevin Walker Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA

Lan Wang Cambridge University, Cambridge, UK

Wei Wang Language Weaver, Los Angeles, CA, USA

Wen Wang SRI International, Menlo Park, CA, USA

Zhiqiang (John) Wang Fair Isaac Corporation, San Diego, CA, USA

Todd Ward IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

Ralph Weischedel BBN Technologies, Cambridge, MA, USA

James V. White BAE Systems, Burlington, MA, USA

William Wong Language Weaver, Los Angeles, CA, USA

Phillip C. Woodland Cambridge University, Cambridge, UK

- Dekai Wu** The Hong Kong University of Science and Technology, Hong Kong, China
- Wei Wu** University of Washington, Seattle, WA, USA
- Zhaojun Wu** The Hong Kong University of Science and Technology, Hong Kong, China
- Eric P. Xing** Carnegie Mellon University, Pittsburgh, PA, USA
- Jia Xu** RWTH Aachen University, Aachen, Germany
- Jinxi Xu** BBN Technologies, Cambridge, MA, USA
- Nianwen Xue** Brandeis University, Waltham, MA, USA
- Sibel Yaman** International Computer Science Institute, Berkeley, CA, USA
- Jin Yang** SYSTRAN Software, Inc., San Diego, CA, USA
- Yiming Yang** Carnegie Mellon University, Pittsburgh, PA, USA
- Yongsheng Yang** The Hong Kong University of Science and Technology, Hong Kong, China
- Kai Yu** Cambridge University, Cambridge, UK
- Dalal Zakhary** Linguistic Data Consortium, University of Pennsylvania, Philadelphia, PA, USA
- Alex Zamanian** BBN Technologies, Cambridge, MA, USA
- Rabih Zbib** BBN Technologies, Cambridge, MA, USA
- Richard Zens** RWTH Aachen University, Aachen, Germany
- Bing Zhang** BBN Technologies, Cambridge, MA, USA
- Pengyi Zhang** University of Maryland, College Park, MD, USA
- Shilei Zhang** IBM China Research Lab, Beijing, China
- Ying Zhang** Carnegie Mellon University, Pittsburgh, PA, USA
- Bing Zhao** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA
- Sherry Zhao** International Computer Science Institute, Berkeley, CA, USA
- Jing Zheng** SRI International, Menlo Park, CA, USA
- Imed Zitouni** IBM T. J. Watson Research Center, Yorktown Heights, NY, USA

Contributor affiliations are as of when the work described in this book was performed.

Chapter 1: Data Acquisition and Linguistic Resources

Editors: Stephanie Strassel, Caitlin Christianson, John McCary, William Staderman, and Joseph Olive

1.1 Introduction

Author: Stephanie Strassel

All human language technology demands substantial quantities of data for system training and development, plus stable benchmark data to measure ongoing progress. While creation of high quality linguistic resources is both costly and time consuming, such data has the potential to profoundly impact not just a single evaluation program but language technology research in general. GALE's challenging performance targets demand linguistic data on a scale and complexity never before encountered. Resources cover multiple languages (Arabic, Chinese, and English) and multiple genres -- both structured (newswire and broadcast news) and unstructured (web text, including blogs and newsgroups, and broadcast conversation). These resources include significant volumes of monolingual text and speech, parallel text, and transcribed audio combined with multiple layers of linguistic annotation, ranging from word aligned parallel text and Treebanks to rich semantic annotation.

The GALE Program's ever-evolving technology goals have demanded a robust, flexible, scalable data pipeline. Over the life of the program, resource creation has become more efficient and adaptive, with increased emphasis on automation and utilization of emergent GALE technology to improve and augment the data creation process. Since the effort involved in linguistic resource creation is significant, coordination is critical. So that data tagged in one task is readily available for exploitation in the next, GALE required implementation of an end-to-end annotation strategy resulting in selected data being consecutively transcribed, translated, word-aligned, and Treebanked, made available for semantic annotation, then tagged for information content under the distillation task. This pipelined approach has not only reduced costs but also offered the opportunity to learn from multiple types of annotation carried out on the same source data. Also, because volume demands are high and resources necessarily limited, considerable effort has been expended to utilize found data and to build upon existing resources wherever possible. Finally, all GALE linguistic data has been developed with broader distribution in mind. This legacy of data has the promise of empowering research for many years to come.

Section 1.2 covers the topic of data resource management, including large-scale broadcast and text data collection. In order to meet the need of performers for more varied, targeted, open-source data against which to test and train their systems, the Linguistic Data Consortium (LDC) devised an unparalleled foreign language media collection

strategy literally spanning the globe, with collection sites in the United States, North Africa, Asia, and elsewhere. Beyond data collection, LDC has managed distribution of hundreds of GALE corpora to dozens of sites, ensuring appropriate protections on blind test data, releases of data designated for official use only, and other restricted data sets, and implementing new approaches to enable sharing of resources within and across GALE teams. Creation of these many resources has also required new technical solutions, including infrastructure for data management, collection, annotation, and distribution.

Section 1.3 covers the wide range of human annotation tasks applied to the data collected under GALE. Human annotation employed under GALE ranges from simple to complex and from well-established to novel:

1. Integrated transcription and translation of training and test data (Section 1.3.1)
2. Parallel corpora manually aligned at the word level (Section 1.3.2)
3. Distillation queries and answers (Section 1.3.3)
4. New approaches to Treebanking (Section 1.3.4)
5. Integrated syntactic and semantic annotations, including parse trees, propositions, coreference, names, and word senses (Section 1.3.5).

Section 1.4 includes reports on the use of algorithms for automatic annotation of data. This topic is explored with regard to rich transcription of speech, segmenting and parsing Arabic, Chinese word segmentation in the context of parsing, and parsing spoken Chinese.

1.2 Data Collection, Distribution, and Management

1.2.1. Large Scale Multilingual Broadcast Data Collection to Support Machine Translation and Distillation Technology Development

Authors: Kevin Walker, Christopher Caruso, and Denise DiPersio

1.2.1.1. Choosing Broadcast Sources

LDC's local collection

In the beginning of the GALE program, LDC's priority was ramping up collection to meet program targets, i.e., 1000 hours each of broadcast news (BN) and broadcast conversation (BC) in each of Arabic, Chinese and English.. Later, LDC added additional receivers to the collection system to increase the range of its Arabic collection, responding to sponsor requests for greater representation of programming across the Arabic-speaking region, particularly, the Gulf region and Iraq. A number of Arabic sources were available from free-to-air (FTA) satellites transmitting over the Philadelphia area. LDC designed program surveys of the various sources that ran roughly twice per hour for sev-

eral days, since these types of channels do not maintain scheduling information. LDC also coordinated with Hong Kong University of Science and Technology (HKUST), Mediant in Tunisia and MTC/ELRA in Morocco to obtain additional sources.

LDC's GALE broadcast collection includes programming from the 26 Arabic sources, 13 Chinese sources, and 3 English sources. Table 1.1 shows the combined total number of hours of broadcast programming collected by LDC and remote sites in the first four years of the GALE program through December 2009.

	P1 Hours	P2 Hours	P3 Hours	P4 thru 12/09
1Arabic BN	751	3896	3944	4940
Arabic BC	700	2813	3125	3153
Chinese BN	918	1231	1055	1452
Chinese BC	1242	1484	1427	1729
English BN	558	655	632	511
English BC	1041	1234	1035	1351

Table 1.1: LDC broadcast collection for GALE

LDC's outsourced collection

GALE's goal to create end-to-end language systems meant that the broadcast collection should incorporate programming from as wide a variety of sources as possible. Accordingly, LDC obtained data collection assistance from several groups located within geographical areas of interest with access to targeted satellite and local programming. LDC's broadcast collection programmer designed a portable collection platform that was deployed outside of the continental US.

1.2.1.2. Broadcast Collection System Design and Operation

Part of the design intent driving the development of LDC's broadcast collection system was that it be modular and regularized. That meant that all of the recording nodes should be interchangeable, that filenames and database fields should follow consistent, formal rules and that signal interconnects should be consistent. The receivers feed into an audio/video (A/V) matrix switch so that any source can be routed to any receiver simply by changing an entry in the schedule.

Initial recordings consist of video, stereo audio, and, in the case of English source, closed captions. LDC collects both audio and video data for each recording so that this material can be reusable for a variety of research purposes and because having access to the video portion of a given broadcast aids troubleshooting system functions and makes auditing more reliable, more efficient, and less error-prone. Recordings are typically transcoded to MPEG-4/AVC at 1Mbps shortly after capture.

The collections system is illustrated in the block diagram in Figure 1.1.

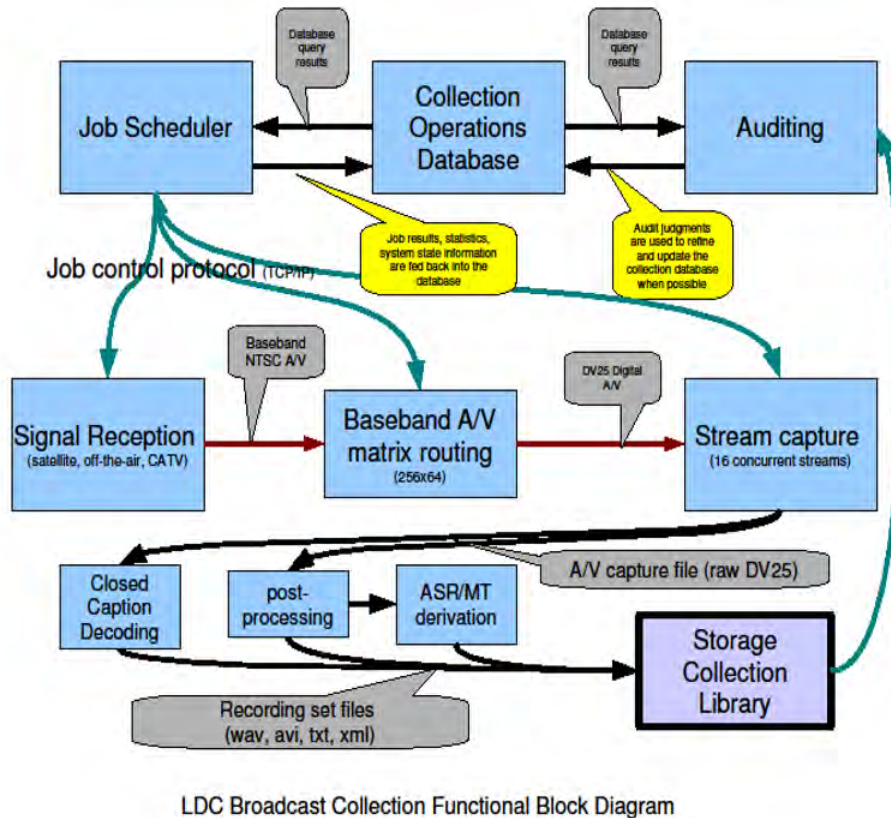


Figure 1.1: LDC Broadcast Collection Functional Block Diagram

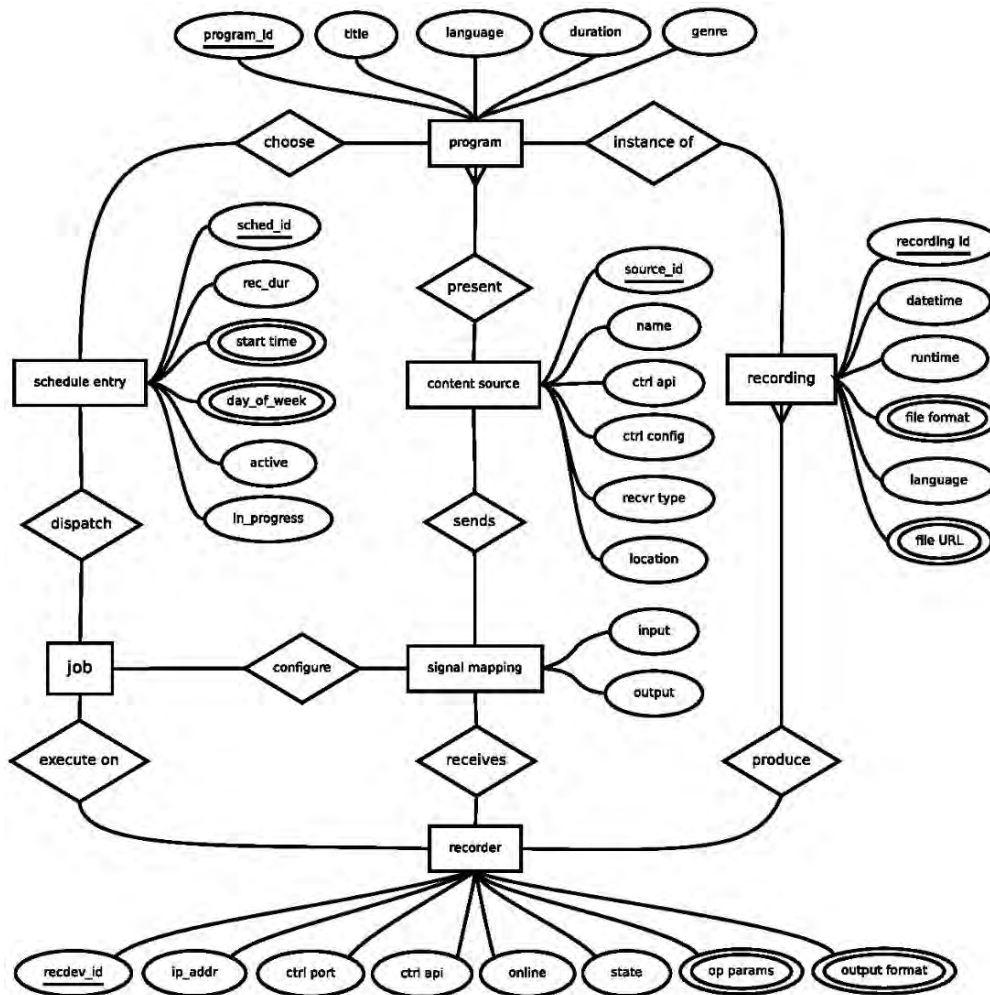
Collection database

The database is the heart of the collection and contains all essential information: it is a history of all of the recordings that have been made; it has configuration and status information for all recorders; it has information about all receivers and associates specific programs of interest with the appropriate receiver; it contains a schedule of all recording jobs that need to be executed, along with their status; and finally, it stores all audit judgments associated with a given recording. Figure 1.2 depicts the collection database.

1.2.1.3. Automatic Speech Recognition (ASR) Technology

LDC has integrated three client ASR systems (from GALE research sites BBN, IBM and SRI) into its daily collection processes for the duration of the GALE program. Transcripts are automatically generated on the BBN and IBM systems for all locally-collected Arabic and Chinese audio data. The text output is used for downstream data selection. Audio data from LDC's remote broadcast collections is not processed daily due to time constraints, but is instead run as needed as part of the GALE evaluation data selection process. Data is processed on the SRI system by request.

Locally-collected broadcasts are automatically pooled onto a centralized server as they are recorded and processed. This server then supplies the extracted audio portions to the different ASR systems which generate transcripts for the previous day's broadcasts.



Broadcast Collection Database Entity Relationship Diagram

Figure 1.2: Broadcast Collection Database Entity Relationship Diagram

1.2.1.4. Portable Broadcast Collection Platform

LDC's portable broadcast collection platform is a digital video recording (DVR) system capable of recording two streams of A/V material simultaneously. It supports analog CATV (NTSC and PAL) and FTA DVB-S satellite programming and is capable of operating outside of the United States. It has a very small footprint and is suitable for transportation as a piece of carry-on luggage. The portable platform and the main LDC collection system share the same code base and rely on a modular, unified hardware specifica-

tion. Improvements in the main collection platform, therefore, translate into benefits for both platforms.

1.2.1.5. Conclusion

Using a highly-automated design and open source utilities, the GALE broadcast collection system represents a significant achievement in delivering volumes of high-quality broadcast data from multiple programming sources and geographic locations. The system has performed impressively. Through GALE Phase 3, LDC had delivered over 11,000 hours of broadcast audio to GALE research sites. A large portion of that data has been transcribed and translated as well, thus providing the GALE community with a considerable quantity of training, devtest and evaluation data.

1.2.2. Text Collection for Volume

Authors: Haejoong Lee, Denise DiPersio, and Robert Parker

1.2.2.1. Web Data Collection

GALE has required huge amounts of web data. With the launch of the program, the goal was to collect two million words for each language. To date, the number has increased to twenty million words for each language. LDC developed a web data collection framework to support GALE. The web collection project for GALE currently collects data from 47,857 weblogs or newsgroups, which are hosted by 94 web sites using the framework.

In this section, we describe the details of the framework, and discuss benefits and limitations of the framework, and a future development plan.

The framework also provides a common ground on which various management applications can be built. For example, a collection management GUI makes it possible for a non-technical user to manage the entire collection process. Another program is capable of producing a real time collection status report out of the tracking database. These tools help streamline the collection process and reduce the management cost.

Data Model

Most weblog and newsgroup sites are structured in a similar way. This structure can be generalized by a three-level hierarchy. At the top, there is a site that hosts individual weblogs or discussion groups. The weblogs and discussion groups form the middle level. Each of the weblogs or discussion groups in turn contains threads of messages, which form the bottom level. In this hierarchy, we call the topmost entities *sites*, middle ones *groups*, and bottom ones *threads*.

The basic data unit of the web data collection framework is a thread. Thread can be viewed as a tree whose root node is either a blog post or an initial newsgroup post. Comments by viewers for weblogs, and replies to a previous post for discussion groups form the inner nodes and leaves of the tree.

As an example, the weblog site *blogspot.com* is a *site*. This site hosts numerous weblogs, which are *groups*, contributed by bloggers. Each weblog has blog posts, e.g. daily coverage of presidential election. Such blog posts become *threads*.

Collection Process

Under the framework, a collection task is defined by a group of sub-processes running in parallel. This includes harvesting, formatting, token counting and language identification processes.

The harvester process identifies and downloads web documents in raw HTML format. Downloaded HTML files are converted into SGML-based format by the separate formatting process.

The token counting process queries the tracking database to get a list of threads that haven't been counted. For each thread in the list, a token counting algorithm is applied to get the token count. The result is the number of white-space-separated tokens for languages like Arabic and English, and number of characters for Chinese. The result is stored in the tracking database.

Language identification is a process of verifying the language of a thread. When a thread is downloaded, a new record for the thread is added to the thread table of the tracking database. By default, the language field of this record is set to the language of the thread's group. It is not uncommon for a group to have threads of different languages. For example, some groups have both Arabic and English threads. Also, it is possible that the language initially assigned to the group is wrong. Thus, it's necessary to have a process to verify the language of downloaded threads. For language identification, a language model trained with character n -grams ($1 \leq n \leq 5$) of the target language is used. For a given document, the language model returns a floating point number indicating the likelihood of the document being in the language in question. This number is stored in the tracking database.

Note that the same token counting and language identification programs can be used across different sites. On the other hand, each site has its own harvester and formatter. Thus, a collection process usually consists of one token counting process, one language identification process, and several harvester processes with the same number of formatter processes, all of which running in parallel.

1.2.2.2. Newswire Collection

As of the commencement of GALE in 2005, LDC was collecting newswire in the target languages from six Arabic sources, two Chinese sources and five English sources. In late 2006, new web sources were added to support GALE (three additional Arabic and four additional Chinese sources).

These new resources represented a 50% increase in LDC's Arabic collection and a 56% increase in the Chinese collection. The average monthly volumes represented by the GALE newswire collection from Phase 2 forward are 13.4 million words/month for Arabic, 64.4 million characters/month for Chinese and 35.3 million words/month for English.

Most of LDC's GALE newswire sources are harvested from provider websites, but some are received via satellite modem, FTP uploads and downloads, NNTP (Network News Transfer Protocol), proprietary clients and email.

1.2.2.3. Future Work

LDC developed the web data collection framework during the early months of the GALE program. Since then, it has been driving the web data collection project for GALE. At the same time, it has been debugged, modified and improved. However, there is still large room for improvement.

First of all, extracting information from the input HTML data is done by harvester and formatter code crafted by human. This is time-consuming and sometimes becomes a bottleneck during an early stage of a collection project. As an attempt to solve this problem, automatic pattern learning algorithms such as (Pennman 2009) and (Holovaty 2007) are being studied.

Another area requiring improvement is to automate the collection process, which currently requires human intervention on a regular-basis. Human intervention is needed because the harvester processes are often confronted by various types of errors such as a bug in the script, temporary problem of a web server, a system problem or a formatting problem of the HTML data it was trying to download, etc.

Although the time required to resolve such issues is minimal, automation that performs analysis of errors followed by an appropriate action is desired for smoother data collection process. Moreover, instead of harvesters being launched manually by a human, a scheduler is being researched that initiates harvester and formatter processes just as frequently as indicated by data volume that sites or groups produce. LDC is planning to add these capabilities for fuller automation of the web data collection process.

LDC continues to work to refine its newswire procedures. This is particularly evident in the evolution of the collection and processing of newswire data from the web. Initially, the web collections were very simple bulk fetches from provider websites, resulting in the collection of duplicate or unwanted content. Over time, LDC has developed more intelligent web-crawling processes to reduce these issues, as well as to exploit newer technologies such as RSS (Rich Site Summary) and ATOM syndication. In addition, LDC has instituted automated reporting to monitor the collection and conversion processes and to provide alerts for any discrepancies.

1.2.2.4. Conclusion

LDC has collected a very large volume of text data for the GALE program. The amount of text data LDC collects and processes each month is 242.8 million words: 151.2 million words of weblog and newsgroup data, and 91.6 million words of newswire data.¹ Robust and efficient text collection infrastructures have been crucial for such a huge achievement. The web data collection framework has provided a scalable and cost-effective method for collecting web data for GALE. LDC continues to use the newswire collection infrastructure for GALE with great success. It was developed in early 1990's

¹ For Chinese data, 1.5 characters were counted as 1 word.

and is evolving ever since. Further research and development for improvement on both the web data collection framework and newswire collection infrastructure are in progress.

1.2.3. Distribution of GALE Resources

Author: Denise DiPersio

1.2.3.1. Introduction

The particular challenge posed by the GALE distribution objective – to distribute significant volumes of data timely and efficiently while conforming with external licensing and contractual arrangements – required an organization with expertise in distributing a broad range of resources to many users. LDC was particularly well-suited for this task; prior to GALE, it had distributed tens of thousands of corpora on media and via web download under various licensing schemes to thousands of users worldwide.

The GALE distribution objective required the performance of five principle tasks: creating a licensing structure; developing a data distribution plan for each GALE phase; preparing the data for release; distributing the data sets; and establishing a Secure Copy Server (SCP) server for rapid sharing of copyrighted data within GALE teams. Each of these tasks is described below.

1.2.3.2. Licensing

LDC developed evaluation license agreements stipulating that GALE participants would have to agree to use the data provided by LDC solely for GALE-related purposes.

1.2.3.3. Data Distribution Plan

GALE established ambitious goals for data collection, creation and distribution: 2000 hours each of broadcast news and broadcast conversation plus five million words of newsgroups and blogs a large portion of which was transcribed, translated and treebanked in each of Arabic, Chinese and English each year. In order to assure that GALE sites received a steady flow of data for training and testing purposes, LDC developed a plan to provide quarterly data deliveries. In response to community feedback, LDC changed that strategy to provide two large training data releases timed to be of maximum value to the sites as they prepared for scheduled evaluations.

At the commencement of the program, LDC provided a kickoff release to GALE research sites that included existing publications as well as previously unreleased data relevant to the program. In the first year, new data was distributed in four quarterly releases. Starting in the second year, training data releases were front-loaded at the request of GALE sites in order to allow as much time as possible to prepare for the evaluation task. In practice, this meant making two large distributions per phase, typically in the fall and spring.

Midway through year 4, LDC had distributed nearly 7000 copies of 509 distinct corpora to 45 sites in 7 countries. That data includes 159 corpora from the LDC catalog designated as being relevant to GALE and approximately 200 new corpora. The latter consists of various training, test and development data (“ecorpora” or corpora designated for an

evaluation community such as GALE), FOUO corpora (“G-corpora,” only for use in the GALE program), restricted corpora (“R-corpora,” data whose distribution is confined to a limited group) and donated data.

1.2.3.4. Data Preparation

GALE task managers and programmers prepared the data, and conducted task-specific data quality checks for content and format before it was mastered for release. The Publications Group performed additional quality assurance that included: checks for file integrity, permissions and file access; checking character ranges and parsing XML in text files; and checking headers in audio files. For media releases, each duplicate was automatically compared with the master using md5 checksums, and random manual checks are conducted on duplicated media. For web download releases, the Publications Group verifies content and completeness before preparing the download.

1.2.3.5. Data Distribution

There were two principal methods for distribution depending on the size of the dataset. HTTP distributions were downloadable from LDC’s members’ web site by authenticated GALE participants. Media distributions were shipped to all eligible participants using established courier services.

1.2.3.6. SCP Server

The SCP server was established in year one in response to requests from sites to allow for rapid sharing of copyrighted data and annotations thereof within GALE teams at different geographic locations. Such activity was otherwise prohibited by the GALE license agreement and by LDC’s agreements with data providers, that is, data could only be shared within one organization at a single geographic site. For security reasons, access to the server was by means of public/private key pairs only.

1.2.3.7. Conclusion

GALE has conferred a measurable benefit to the data needs of the linguistic research community. Most of the data collected and/or created for use in GALE will be released in the LDC catalog. As of this writing, some year one materials were already available, and other data was in the pipeline. This is consistent with LDC’s mission to share linguistic resources as broadly as possible.

Furthermore, GALE’s volume and distribution requirements provided the acid test for existing processes and have resulted in the establishment of an efficient infrastructure for data distribution. This includes the creation of stable master copies, media shipment tracking, perseverance of downloadable releases and rapid data sharing via the SCP server. That infrastructure can now be propagated to other common task projects with relative ease.

1.2.4. Technical Infrastructure Supporting Large-scale Linguistic Resource Creation

Authors: Kazuaki Maeda, Andrea Mazzucchi, and Christopher Cieri

1.2.4.1. Introduction

The DARPA GALE program set a new level of technical requirements for LDC due to the quantity and diversity of training and evaluation data, including the manual and automatic transcripts, manual and automatic translations, found parallel text, word aligned parallel text, distillation annotations and treebank annotations. This section provides an overview of the technical infrastructure supporting the GALE program at LDC.

1.2.4.2. Data Storage Infrastructure

The data storage requirements for GALE included large storage capability, flexible expansion capability and solid data security. The size of the dynamic data, such as annotations, software and documentation, exceeded more than 1TB for each phase of GALE, which is more than 10 times larger than previous resource creation projects such as TDT (Cieri *et al.* 2002). The size of the static data, such as audio, video, newswire and web text data, is even larger. The total hours of broadcast news collection per phase, which includes both video and audio data, grew from approximately 5,000 hours in Phase 1 to more than 11,000 hours in Phase 3. The volume of web text collection grew from almost none before the GALE collection to approximately 7 million threads in Phase 4. The total size of the data storage for GALE has reached 90 TB as of this writing.

1.2.4.3. Software Infrastructure

The challenges presented by large-scale GALE data creation necessitate software tools that maximize efficiency. Most of GALE data collection, annotation, and distribution tasks use custom-built software developed by LDC's technical staff. Software tools are written to support annotation according to guidelines and other specifications, and to output consistent data formats.

Annotation Tools: In order to maximize the efficiency of annotation software development, we have adopted a common set of technologies for stand-alone annotation tools, such as Python, Qt, PyQt and MySQL, which are compatible with those previously developed at LDC, such as AGTK (Maeda *et al.* 2006). This allows the developers to reuse software components, and allows flexibility for allocation of software developers to available tasks as all developers are familiar with these technologies. Other tools use web-based technologies, such as PHP and Java Script. Annotation tools developed for GALE include XTrans, the Translation QC Tool, the SU annotation tool, the Distillation Annotation Toolkit, the broadcast audio auditing tool, and the GALE evaluation snippet selection tool.

Annotation Workflow Management Tools: LDC has developed a custom annotation workflow management system called AWS (Annotation Workflow System), used for

most in-house manual annotation tasks at LDC. LDC has also developed a workflow management system, specifically for the Machine Translation (MT) Post-Editing task. The MT Post-editing system is web-based, allowing MT Post Editors to log in, receive their current assignments, annotate them, and submit them via the system (Maeda *et al.* 2008).

Parallel Text Alignment Tool: LDC has developed two toolkits that are used for automatically aligning parallel text for GALE. One is the BITS (Bilingual Internet Text Search) toolkit (Ma and Liberman 1999). The document alignment module of this toolkit was used for the document aligning task. The other toolkit is called the Champollion toolkit (Ma 2006). This toolkit was designed for sentence alignment of noisy data, and proved to be very useful for alignment of parallel text harvested from web sites (Maeda *et al.* 2008a).

Content Duplication Identification Tool: It is inevitable to have some degree of content duplication in the raw collected data. For example, newswire agencies often distribute multiple versions of the same article with minor updates; the same clips may be used in multiple broadcast programs. LDC runs a content duplication identification tool on the evaluation snippets, and on the source files selected for translation training data.

Language ID Tool: A language identification tool is run over LDC's web text collection to confirm that the majority of the collected text is in the target language. This is important for multilingual sources and noisy sources, such as newsgroup text, in which postings in non-target languages are often observed.

Arabic Morphological Analyzer: The LDC Standard Arabic Morphological Analyzer (SAMA), which is based on the Buckwalter Arabic Morphological Analyzer (BAMA) (Buckwalter 2004), is run on the input data to the Arabic Treebank annotation task. The BAMA tool was developed by former LDC staff member Tim Buckwalter, and has been extended as the SAMA tool. The SAMA tool is extensively used in LDC's Arabic language projects.

1.2.4.4. Training Data Creation

LDC is responsible for creating the following types of training data for GALE:

- Broadcast
- Web Text
- Transcription
- Parallel Text
- Distillation
- Word Alignment
- Arabic Treebank
- English Translation Treebank

1.2.4.5. Evaluation Data Creation

LDC also creates development and evaluation data for GALE MT evaluation programs. While the basic data creation process is the same for the training data, the evaluation data creation requires more detailed quality control and secure handling of the data. The evaluation data process is done in close collaboration with the National Institute of Standards and Technology (NIST), which administers the GALE MT evaluation.

MT Evaluation Snippets: Unlike the training data, the GALE MT evaluation uses snippets of documents or broadcast recordings. The evaluation snippets are created in the following steps.

1. Data collection
2. Program-level broadcast audit
3. Decision of evaluation epoch by GALE data committee
4. Manual selection of evaluation snippets
5. Transcription (for BC and BN) and SU annotation (for NW and WB)
6. Duplicate content identification
7. Down-sampling of evaluation pool by NIST
8. Initial translation
9. Complete QC of source and translation
10. Final selection of evaluation snippets by NIST

In each stage above, snippets that are found problematic or unsuitable are dropped from the data set, so over-selection of data is crucial. For the Phase 4 evaluation, LDC selected four times as much data for the initial pool than remained in the final evaluation data set.

Data Security: All GALE evaluation data is protected with strict data security control. Evaluation data on the file servers is protected with Unix group access control, and only LDC personnel who are working on the GALE program are permitted to be in this restricted group.

1.2.4.6. Conclusion

The technical infrastructure described in this section represents the work done for the GALE program, which is the largest scale language resource creation effort in the history of LDC. This infrastructure was built on our experience in GALE as well as other projects at LDC, and will be a fundamental resource for future work at LDC.

1.3 Human Annotation

1.3.1. Linguistic Resources for Transcription and Translation

Authors: Meghan Lammie Glenn, Lauren Friedman, Stephanie M. Strassel, Zhiyi Song, Gary Krug, Kazuaki Maeda, Haejoong Lee, and Christopher Caruso

1.3.1.1. Introduction

The GALE program required significant volumes of new training data -- hundreds of hours of transcribed and annotated speech, plus hundreds of thousands of words of parallel text -- for each language and genre being evaluated in the program. As such, LDC's initial focus was on developing efficient, scalable processes for data selection, transcription and translation. The primary emphasis was on quantity and coverage, especially with respect to training data for Machine Translation (MT) and Automatic Speech Recognition (ASR) systems. Large volumes of Arabic and Chinese text and audio were collected to provide background language models and data for unsupervised training, and a sizeable portion of the collected data was then selected for manual transcription and translation. Over time, as the archive of existing training data expanded and GALE system performance improved, requirements for new linguistic resources evolved accordingly, shifting from rapid, high volume resource production to a more refined and targeted selection of material to address specific gaps in coverage. Each stage of the GALE program has motivated advances in LDC's resource creation pipeline. New annotation tasks and guidelines, quality control procedures and enabling infrastructure like annotation software and data formats have been developed at regular intervals to keep pace with evolving program requirements. As GALE system performance improves, LDC has increasingly relied on ASR and MT system output to augment and improve the data production pipeline.

This section describes LDC's methodology for creating linguistic resources for transcription and translation, covering both training and evaluation data. We describe data selection; annotation including transcription, translation, and pilot efforts; quality control practices; and the novel technical infrastructure developed to support the unique demands of GALE.

1.3.1.2. Training Data Production

Data Selection

Although some transcribed audio and parallel text may be harvested online or adapted from existing resources, the bulk of the training data developed for GALE is created manually. Training data production starts with selection of material for manual transcription and translation. This material is chosen from Arabic and Chinese source newswire, web text, broadcast news and broadcast conversation collected by LDC for GALE, following procedural guidelines developed by LDC in consultation with NIST, DARPA and GALE research teams. Detailed volume and genre requirements for each round of selection are established by LDC and GALE sites, and refined at regular intervals to meet evolving program goals.

For each batch of data to be transcribed or translated, a selection pool is prepared that adheres to requirements for (1) source and program variety, (2) genre balance, (3) broadcast/publication dates, and (4) content of existing transcript and translation resources. The selection pool is heavily weighted to prefer sources and programs that are underrepresented in previous training data releases. Priority is also given to recently collected data and/or material from epochs close to the current evaluation epoch. Once the selection pool is established, human annotators quickly review each candidate, excluding items that do not meet requirements. Because GALE is primarily focused on transcription and translation of Modern Standard Arabic (MSA) and Mandarin Chinese, candidates that contain large volumes of non-MSA or non-Mandarin Chinese dialects are typically excluded from selection. Similarly, material whose content focus is deemed inappropriate - coverage of sporting events, spam or other offensive material, soap operas, and the like - is also excluded.

New Selection Methodology

After substantial volumes of high-quality, broad coverage training data for translation and transcription became available during the initial phases of the program, there was growing interest from GALE research sites and LDC alike in developing refined data selection techniques that would especially target high-payoff data; i.e., novel material that does not duplicate the salient features of existing training data, and material that is especially difficult for ASR and MT systems. In response to this need, LDC began working closely with research teams to develop a series of “smart data selection” techniques to guide selection of new transcription and translation training data. These new approaches rely both on understanding the properties of existing training data (for instance, what n-grams already occur in sufficient numbers in existing parallel text?), and on detecting data on which system performance is unexpectedly poor.

Smart Data Selection for Transcription

As a starting point, LDC and GALE research teams collaborated to install production MT and ASR systems at LDC, and to integrate these systems into LDC's regular collection pipeline (Section 1.). Collected audio recordings are run daily through the ASR systems, yielding two dozen or more hours of ASR output per language per day. GALE researchers worked with LDC to develop a process that could leverage this resource by tapping into a simple intuition: if multiple ASR output streams have high rates of agreement with one another for a single recording, ASR error rates for that recording are probably low, whereas low agreement rates likely signal high ASR error rates. Audio recordings showing moderately high rates of disagreement among ASR systems could then be targeted for manual transcription, to produce high-value training resources when compared to resources selected using traditional methods.

To implement the new approach, segmented ASR output for Arabic and Chinese is first stripped of all segmentation markers and converted into plain text for more accurate comparison. The resulting files are then processed by *sclite* (Fiscus *et al.* 2006), a scoring script developed by NIST, which generates a report of the word error and agreement rates among the different systems for each show's recording. The result of the process is a list of programs ranked by scoring agreement for all programs in a given selection pool.

A pilot study conducted by LDC during GALE Phase 3, which compared the output of two ASR systems, demonstrated a “sweet spot” between 30% and 80% agreement. Annotators audited the ranked list of audio files and recorded information about each file, noting lengthy music segments, foreign language speech, non-native speakers, commercials, and the like. Recordings with 30% or lower agreement tended to contain extended commercials, music programs, heavy regional speech variation, or recording errors. Those with 80% or higher agreement tended to be “talking-head” style read speech, usually by a single speaker. LDC leverages this information to exclude files from the two extremes during selection. The programs between the 30% and 80% error range may contain speech phenomena for which there is not yet enough training data for ASR systems.

Once the outlier programs have been removed from the selection pool, annotators review the programs to further eliminate those which may not be suitable for GALE transcription and translation due to their content or level of speaker interaction. The files approved during this process are then manually transcribed.

An added benefit to leveraging ASR technology and scoring recordings according to the level of agreement among shows is that programs with exceptionally low or high levels of matching segments may be further examined to determine the causes of variation, enabling LDC to identify potentially problematic audio elements of broadcasts. Another potential use for these reports would be to more quickly identify “challenge” sets of broadcast genres for system development. This ASR-assisted selection approach has been implemented to select half of all new training data transcripts.

Smart Data Selection for Translation

Starting in Phase 3, data selection for translation underwent a similar transformation. Whereas previous selection methods produced full-document translations (a whole newswire article, blog post or broadcast story), the new method targets selection of individual high-yield sentences -- where high-yield is defined having features that are novel compared to existing stores of training data. LDC collaborated with GALE research teams, in particular IBM and SRI, to develop scripts to implement the novel selection method. The process begins by establishing a table of n-grams for sentences in existing training data sets. Next, LDC prepares a pool of candidate documents for a given language and genre. The script is run over this data pool, and generates a list of candidate sentences, scoring each according to its uniqueness against the table of existing n-grams. Humans quickly scan the resulting ranked list of candidate sentences, discarding inappropriate material prior to manual translation. Initial feedback from GALE teams is positive, and this sentence-based method is now exclusively used to select new translation training data.

1.3.1.2.1. Training Data for Transcription

Manual Transcripts

While some manual transcription and related speech annotation activities are performed by local LDC staff, we also maintain relationships with a number of international partner sites and commercial agencies who possess the infrastructure and know-how to rapidly produce high-volume, low-cost training transcripts and translations of sufficient quality. All transcription partners undergo regular hands-on training conducted by LDC

annotation staff, and all partners utilize LDC infrastructure including transcription guidelines and user interfaces, to ensure consistent quality. Moreover, small volumes of data are regularly subject to dual, independent transcription to provide material for analysis of human transcription variation and to ensure that inter-transcriber agreement both within and across sites falls within an acceptable range.

Transcription Task Specifications

All transcribers, whether they are working at LDC or at one of our partner sites, are required to follow formally-defined transcription specifications to ensure efficiency and consistency among all transcripts, regardless of source language. Guidelines differ depending on the purpose of the data, though the goal for all transcription approaches is essentially the same – to produce a verbatim transcript with speaker and story identification in the most efficient manner. Training data requirements place a higher value on procuring large volumes of transcripts with less emphasis on gold standard perfection, whereas evaluation data requirements reverse these priorities. This section describes the training data transcription approaches employed within GALE.

Quick Transcription

Quick transcription (QTR) for broadcast news and broadcast conversation aims to quickly produce a (near-) verbatim transcript, time-aligned at the phrase or pause-group level, with markup limited to story boundaries and speaker turns/IDs. The emphasis is on efficiency, with a goal of producing fairly accurate transcripts in five to ten times real-time (in other words, five to ten hours of transcription effort per hour of speech). QTR is used only for training data production.

Quick Rich Transcription

Quick-Rich Transcription (QRTR) was defined at the outset of the GALE program and is the default manual transcription method for GALE. The goal of QRTR is to balance efficiency and completeness, producing transcripts that are maximally useful for downstream manual and automatic annotation tasks including translation, distillation and Treebanking. Like QTR, QRTR results in verbatim, time-aligned transcript with story boundaries and speaker turn identification, but QRTR also incorporates features like sentence-unit (SU) identification and sentence-based time alignment. (LDC 2008).

The methodology for identifying sentence-units in spontaneous speech grew out of work performed during the metadata extraction (MDE) evaluation of the DARPA EARS program, which aimed to annotate verbatim transcripts of spontaneous speech in order to make them maximally readable (LDC 2004). Annotations included removing disfluent non-content words such as filled pauses or backchannels, and labeling each SU according to its type, such as statement, question, or incomplete utterance.

QRTR incorporates SU annotation into the audio segmentation process, which is the first stage of transcription. A sentence-unit (SU) is a natural grouping of words produced by a single speaker. SUs have semantic and syntactic cohesion. When creating SU boundaries for spoken language, the transcriber's goal is to identify a semantically and syntactically cohesive group of words that constitute a reasonable sentence-like unit. The segmentation remains consistent for all downstream tasks, permitting a mapping from the

source audio signal to the source transcript to the English translation and annotations for each sentence-unit. Transcription rates for QRTR are targeted at 15 to 20 times real-time.

1.3.1.2.2. Overview of QRTR Rules

Transcription rules are consistent for all three source languages. The elements of a quick rich transcript include: verbatim transcription, time-aligned story boundaries; speaker turn, section and sentence-type identification; speaker identification, and standard treatment of common spoken phenomena (e.g., hesitation sounds, spoken letters or numbers, and disfluencies).

Segmentation

Transcription begins with audio segmentation, which involves marking structural boundaries including story boundaries, speaker turns and sentences. Speakers are identified by name where possible, or by a unique identifier, such as “speaker1”. Transcribers also indicate the speaker’s type – male, female, child, or other – and native or non-native speaker status. Story boundaries are inserted when there is a change in topic, and stories are labeled according to their style: talking-head broadcast news sections are labeled “reports”; discussions or informal chats are labeled “conversations”. Musical interludes, commercials, public service announcements, and periods of silence are left unsegmented, or in some instances may be segmented and labeled “non-trans” and left untranscribed.

Verbatim Transcription and Orthography

Once audio has been segmented into smaller units, annotators transcribe the content of each segment. Transcribers use standard Mandarin Chinese, MSA, and North American English orthography. Special conventions are used to flag certain speech phenomena like disfluencies, mispronounced words, and non-target language speech. Quality control checks verify the format of the resulting file, as well, as overall transcript quality.

Foreign or Colloquial Speech

When transcribers encounter speech not in the target language, they identify the language, but do not attempt to transcribe the region.

Colloquial speech regions are transcribed, but are marked as being “non-standard.” Due to the prevalence of dialectal speech in Arabic broadcast conversation files, transcribers mark speech that is not MSA as such, for example:

```
<non-MSA> </non-MSA>
```

When the transcriber understands the dialect, he/she transcribes the utterance, for instance:

```
<non-MSA> كمان </non-MSA>
```

While transcribers do not label specific Arabic dialects during the transcription process, identifying non-MSA regions at this stage may help downstream processing,

error analysis, or setting the stage for Arabic linguists to revisit those regions as a separate dialect annotation task.

Web Transcripts

To supplement manual transcription efforts, LDC routinely harvests available web transcripts in Arabic, Chinese and English for sources that are part of our ongoing audio collection, and conducts periodic searches to identify transcripts from new broadcast sources.² Plain text transcripts are extracted from HTML files, converted to UTF-8, and divided into sentences based on punctuation characters. While web transcripts are typically verbatim or near-verbatim, the format differs considerably from LDC's native transcription format in several ways. First, web transcripts are not time-aligned with the audio; second, they do not follow LDC-style transcription conventions; finally, story boundaries and sentence units (SUs) have not been explicitly labeled.

Transcription Volume

LDC has released over 10,000 hours of training transcripts in Arabic, Chinese and English from Phase 1 through Phase 4 of GALE. The total number of transcript hours released, and the distribution of those transcripts by language and genre, varies from phase to phase depending on the specific requirements expressed by the GALE research teams, and also in part due to the availability of collected and audited broadcast data.

Language	Genre	Phase 1	Phase 2	Phase 3	Phase 4
Arabic	BC	454	798	667	459
	BN	640	751	557	171
Chinese	BC	975	478	280	262
	BN	586	492	341	310
English	BC	763	128	154	282
	BN	507	122	151	65

Table 1.2: Volumes, in hours, of manual and web training transcripts distributed through Phase 4 of GALE.

1.3.1.2.3. Training Data: Translation

All statistical machine translation systems require significant linguistic resources for system training; GALE required LDC to produce these materials on a larger scale and with greater variety and agility than ever previously undertaken. In response, it was necessary for LDC to develop novel methods, technical infrastructure, and best practices to create a pipeline that is flexible, scalable and efficient. While the quality of training data is important, it must be balanced against the unwavering demand for high volume, low cost, and rapid distribution -- in well-established translation genres like newswire, but also in diverse and challenging new genres like talk shows and weblogs. As the GALE program progresses, the translation pipeline must also become more refined, and the efficiency of the processes and the quality of the data improve in tandem. This section fully

² LDC's web collection efforts are described in detail in Section 1.2.2.

describes the translation pipeline, which includes not only the actual translation process, but also numerous preparation and quality control measures that shape the final product.

Before Translation

LDC conducts multiple data preparation stages before sending source documents to translation agencies. First, all files put through the translation pipeline are either manually selected or – if automatically selected – manually reviewed to reject non-target content, as previously discussed in the Data Selection section.

Corpus-wide Scans

Once the translation pool has been identified, the corpus is automatically scanned to remove duplicate documents and to ensure that there is no overlap between the training and evaluation sets. This crucial step further focuses the use of resources on novel data and is one of many safeguards to preserve the integrity of evaluation sets.

SU Annotation

Selected data is segmented into SUs. SU segmentation is performed manually or automatically depending on the genre and timeline. In general, newswire data can be automatically segmented; QRTR transcripts are manually segmented during transcription; QTR transcripts and web documents must be manually segmented prior to translation. Automatically-segmented data is manually checked and corrected to confirm that logical semantic segments have been created. SU segmentation is preserved throughout the training data pipeline.

Pre-processing

Before the data pool is outsourced for translation, it is subsampled in order to achieve the best diversity and balance among sources and genres, according to requests or requirements for each dataset. The final set of segmented source data is then sent out in a special UTF-8 text format where each numbered segment of source data is paired with a corresponding blank numbered line. For example, an outgoing Arabic source file would look like this:

```
<ar=1> Arabic text
<en=1> [blank line]
<ar=2> Arabic text
<en=2> [blank line]
```

Translators perform the translation one segment at a time, inputting the English on the blank line for each source segment. This process was developed to create perfect alignment at the sentence level between the source and translation. Using this translation format, manual or automatic alignment at the sentence-level is not needed downstream; the translation is aligned as it is generated.

These translator-ready files are then collected into “kits” for outsourcing to vetted commercial translation agencies. Kits are customized for each agency based on target

volume, agency expertise, file length, and level of difficulty. Different sources are distributed evenly across agencies so that any observations about a particular source are not invalidated by different agencies' variability in style or quality.

Manual Translation

In addition to the technical infrastructure it has developed, LDC has also produced extensive documentation to systematize and standardize large-scale human translation for the GALE program. While variability cannot be completely eliminated in a human task such as translation, where there will always be multiple correct translations, it is controlled as much as possible through the provision and enforcement of detailed and rigid translation guidelines developed for GALE, which instruct translators to emphasize accuracy and fidelity to the source text, and provide clear guidance on how to approach special cases (LDC 2009).

These guidelines include instructions and examples to aid translators in approaching documents collected for GALE, which differ significantly from the data translators usually encounter, as well as, requirements for how and by whom documents must be translated. LDC uses multiple translation teams for each language, and each team must have at least one translator native in the source language and one native in the target language.

Translators are required to follow the guidelines' specifications for translating speech disfluencies, factual errors, filled pauses, proper names, and more. Instructions for handling potential challenges – like typos, neologisms, emoticons, and other features of web data – are also included. Addressing these special cases in the translation guidelines assures consistency where there would otherwise be variability if individual translators relied on their own best judgment. Furthermore, clear markup that indicates typos, translator uncertainty, and made-up words allows sites and evaluators to treat these instances differently when necessary.

The translation guidelines are a living document, constantly updated to include more examples and address new translation issues. Releases include READMEs to indicate which version of the guidelines has been applied to each particular data set. Distinct guidelines have been developed for Arabic and Chinese source languages so that language-specific issues can be adequately addressed. Translation agencies are always required to use the most up-to-date version of the guidelines to ensure that all data is translated according to the same standards.

1.3.1.2.4. After Translation

Post-processing and Sanity Checks

Once a set of data has been translated, incoming translations are processed using a suite of scripts that extract the English lines from the merged source/translation documents and perform a bevy of sanity checks. These checks ensure that all files have been returned, all files are correctly encoded, all source segments are present, all segments have corresponding translations, and the data format is intact. Thus, most problems with incoming translations are discovered automatically and can often be rectified automatically as well. Identifying problems at this early stage allows LDC enough time to return files to agencies for retranslation when errors cannot be easily fixed by LDC.

Manual Quality Control

Manual quality control (QC) occurs after this series of automatic sanity checks. Even when translations are created by skilled translators and then proofread closely, erroneous translations and areas of ambiguity frequently exist in raw translations. Certain genres that are included in GALE, especially broadcast conversation and web data, provide particular challenges to human translators, as noted previously. In addition, since translation is not an exact science, two independent translators will typically produce two different translations for an identical source document. However, human translations produced for MT training and evaluation must be exact and fully expressive: the translation must convey no more and no less information than the source.

The quality control processes undertaken for training and evaluation data are vastly different. Evaluation translation quality control is discussed in detail in Section 1.3.1.3. For training data, a subset of each translation delivery (between 10 and 20 percent of the total word count) is checked by bilingual LDC annotators trained in the appropriate procedure. Annotators apply specific scoring mechanisms according to a rubric included in the translation guidelines provided to agencies. Translation errors are categorized as: syntactic, lexical, poor English usage, or typographic (significant spelling or punctuation mistakes). Each category is given a certain number of points. Deliveries that receive a failing score (too many points) are rejected and returned to the agency; payment is withheld until corrections are completed on the entire translation set (not just the files that were reviewed) and the revised translation delivery meets QC standards.

1.3.1.2.5. Training Data: Parallel Text Harvesting

LDC has also developed a set of software tools including BITS (Ma and Liberman 1999) to identify potential parallel text resources among online archives of multilingual documents (Maeda, *et al.* 2008a). LDC uses this software to scan and harvest likely parallel documents from the web on a regular basis. These sources include newswire articles from multilingual news agencies, such as Agence France Presse and Xinhua News Agency.

Parallel text documents on the web come in a variety of formats. After harvesting, files are converted to a text format with a predefined set of SGML or XML markups, and the document mapping module of the BITS system is then run to identify pairs of possible parallel documents. Once pairs are identified, each document is automatically segmented into sentences and then processed by the Champollion sentence aligner (Ma 2006) to create sentence mapping tables. For the GALE program, LDC has created and distributed over 82,000 automatically harvested document pairs containing likely Arabic-English parallel text, and over 67,000 document pairs for Chinese-English.

1.3.1.3. Evaluation and Development Data Production

One of the greatest challenges encountered in producing linguistic resources for GALE is creating the gold standard source and reference sets for evaluation. The GALE evaluation paradigm relies on a carefully constructed test set, which includes careful manual selection and transcription (for audio genres) of the evaluation data pool. On the surface, creation of test data for machine translation is straightforward: take the set of evalu-

ation documents and manually translate them. But, like any task involving human judgment, “translation” is not a monolithic task and there are multiple decision points along the way. As the provider of evaluation data, LDC must consider not only the fully articulated requirements for test data – the type stated in an evaluation plan – but, also hidden assumptions and implicit requirements that are equally important in constructing appropriate data for evaluation.

Data Selection

Evaluation data is selected carefully, according to very specific guidelines. An evaluation epoch is first decided collaboratively by LDC, DARPA and NIST. LDC compiles candidate files from that epoch to construct a pool of newswire, broadcast audio and web data. Each candidate file (audio or text) is examined carefully to identify snippets that meet the criteria specified in the selection task specification. These text and audio snippets typically correspond to a topically cohesive story selected from each document or recording. Information about topic category, presence of dialect, and acoustic and dialect features of the audio file are noted by annotators.

Segmentation and Transcription

Segmentation for evaluation data is especially important, since MT output is scored at the segment level. Poorly constructed segments can lead to ambiguity and then inaccurate MT scores. Segmentation errors that affect the meaning or interpretation of the text are always corrected, even if they are discovered at the final stage of the pipeline.

SUs are manually identified for the newswire and web snippets. Selected audio snippets are carefully transcribed and segment boundaries identified by professional transcription agencies and LDC annotators, following an enhanced version of the Quick Rich Transcription (QRTR) specification. Transcription of evaluation data differs from the training data approach in that it requires multiple manual reviews of a file to produce a gold standard reference transcript. Transcribers first create a basic quick-rich transcript, with speaker and sentence unit annotation and a verbatim transcript. Senior transcribers then review each utterance carefully to verify the transcripts' accuracy, making modifications where necessary. The lead transcriber conducts a final review over all transcripts before marking them complete, and the reference transcripts are then passed to the gold-standard translation process.

Translation and Manual Quality Control

Transcribed and segmented files are reformatted into a human-readable translation format, and assigned to the best professional translators for careful translation, following LDC's standard GALE translation guidelines (LDC 2009).

After translation, reference files undergo several additional stages of annotation and quality control to correct errors, finalize and correct segmentation, improve translation adequacy, add translation variants, standardize proper nouns, verify technical terms and so on, with the ultimate goal of having gold standard translations that are absolutely faithful to the source data in terms of meaning, fluency, structure and style.

In order to standardize the translations and produce an appropriate reference for evaluation, LDC has developed a six-step translation and QC process:

- 1) A source-language dominant bilingual translator produces a preliminary translation emphasizing accuracy;
- 2) A target-language dominant bilingual translator revises the translation to improve fluency;
- 3) A source-language dominant bilingual annotator checks translation for errors and omissions;
- 4) A source-language dominant bilingual senior annotator checks for remaining errors, improves fluency, corrects and standardizes named entities;
- 5) A target-language dominant bilingual annotator improves fluency and adds translation variants where required;
- 6) A target-language monolingual annotator reviews for fluency and consistency, and flags questionable regions.

Steps 1 and 2 are largely the same as those used in the creation of training data, but additional quality control loops are added to meet the evaluation data standards. The translations delivered after Step 2 are considered final and complete by the agencies; the subsequent steps are an above-and-beyond layer added by LDC in order to ensure the highest possible confidence in the released gold standard.

Steps 3 through 6 are performed in-house. In Step 3, the annotator focuses only on correcting egregious errors. The main objective of Step 4 is to resolve any nuanced issues with the translation, while verifying total fidelity to the source – a requirement for the GALE evaluation. In Step 5, fluency problems are corrected and translation variants are introduced to clarify regions of ambiguity.

Relative to Step 5, Steps 3 and 4 are extremely time-intensive. For the most recent GALE evaluation, Step 3 averaged 25 minutes per Arabic document and 21 minutes per Chinese document. (Each document is between 150 and 350 tokens, where a token is an Arabic word or Chinese character.) Step 4 averaged 13 minutes per Arabic document and 15 minutes per Chinese document. Step 5, however, averaged just 6 minutes per document for both languages, since the translation is generally in excellent condition by the time it reaches Step 5.

The final check, Step 6, is a quick but thorough read-through of all of the translations, with an eye to any errors that may have been introduced inadvertently in previous steps. The Step 6 reviewer must also ensure that the translations read as correct, fluent English, independent of the source text. Each stage requires a different level of expertise, and while six different translators might produce six different translations, the six-person translation and QC team – who work independently but consecutively on one working document – is designed to achieve a higher level of consistency and predictability. Step 5, where translation variants are introduced when required, is especially important for ensuring that any alternate, but equally accurate interpretations of the source text are included in the final translation reference.

Development Data

In addition to training and evaluation data, LDC produces development datasets, which are typically small-volume, carefully-created datasets that resemble evaluation

data in terms of preparation and total volume, and are carefully selected from the epoch immediately before the evaluation collection epoch. In terms of production procedures, devtest sets fall somewhere between the carefully controlled process for evaluation data, and the high volume rapid turnaround approach taken for training data production. Requirements for development data vary across phases, and are typically specified by GALE research teams. For instance, in GALE Phase 4 LDC selected, transcribed, annotated, and translated a large devtest corpus of over 600 snippets per language per genre. A smaller subset of each phase's devtest set is typically enhanced with additional annotation including word alignment and multilingual Treebanking.

1.3.1.4. Annotation Tools and Infrastructure

The constantly evolving data requirements of the GALE program have required LDC to develop an innovative suite of supporting software and related technical infrastructure, in order to construct an adaptable and extensible pipeline for transcription and translation that appropriately balances concerns of quality, efficiency, complexity and flexibility.

XTrans

Among the challenges faced in creating transcription manual transcripts for training or evaluation purposes is the efficiency of the transcription software itself, since an efficient tool can boost productivity and provide opportunities for richer annotation or research. Broadcast recordings present the challenge of containing overlapping speech from multiple speakers on a single audio channel. Creating segments and transcripts for such overlapping utterances is a function that many currently available transcription tools lacked. To meet demands for increased volumes of spontaneous speech, which often include conversations among multiple, simultaneous speakers, LDC required an efficient and straightforward tool for transcribing overlapping speech. Furthermore, the need to produce large volumes of transcripts with a consistent file format across multiple languages promoted the development of a new transcription tool.



Figure 1.3: Arabic broadcast conversation transcription session in XTrans.

XTrans, the resulting speech annotation toolkit (Glenn *et al.* 2009), supports a full range of speech annotation tasks including quick and rich transcription and annotation of broadcast audio, telephone speech and meetings. Powered by Qt's international language support, XTrans can be used for transcription tasks in many different languages. It is easily ported to most UNIX derivatives, Microsoft Windows and Mac OS X. XTrans consists

of several re-usable components such as text and waveform display. Most of the components are written in Python with some components written in C++ (LDC 2007).

The Qt text widget also supports bi-directional text input for right-to-left (RTL) languages like Arabic. This functionality allows transcribers to insert English-language metadata tags in the Arabic language transcript. Clicking on a segment in the transcript highlights the corresponding segment in the waveform, and vice versa.

XTrans alleviates the challenge of efficiently annotating multiple overlapping speakers. The tool incorporates the concept of a *virtual speaker channel* (VSC). Each VSC corresponds to one speaker, rather than to any particular physical channel in a sound file. A VSC may also be used to represent background noise or other non-speaker sound sources. The addition of this feature allows transcribers to transcribe a potentially unlimited number of simultaneous speakers without having to resort to any cumbersome markup.

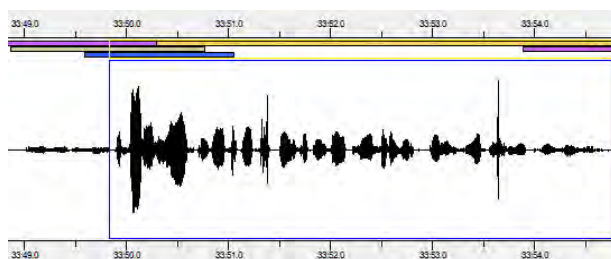


Figure 1.4: Close-up of overlapping utterances in XTrans.

Each speaker in the recording is randomly assigned a color by the tool. The segments for that speaker are coded with the same color in the transcript window. The waveform display also shows the span of each segment in the audio channel as a color-coded box. Overlapping utterances are depicted in the waveform display by stacked boxes, as shown in Figure 1.4. The overlapping segment information is then extrapolated from the times in the transcript file, as shown by the start and end times of the two segments in the following example:

Start	End	SpeakerID	Transcript
394	400	speaker1	那一会儿我们跟您详细聊一聊,给我们讲讲您的故事,好吗,好谢谢何伯伯
394.2	399	speaker2	呃呃,呃呃呃,呃呃呃

Table 1.3: Overlapping segments in plain text transcript.

XTrans incorporates a number of quality control functions that improve the accuracy of speaker identification by the transcriber, such as the ability to listen to random segments or all segments from a selected speaker. In addition, transcribers can listen to all unsegmented audio “gaps” in the transcript, which assists in identifying regions of missed speech. It is easily configured to make transcription efforts more efficient -- for instance, all segmentation functions can be performed using keyboard shortcuts instead of a mouse. Users may also add their own keybindings or modify existing keybindings to suit their comfort and preference.

Translation QCTool

The multi-stage translation quality control process for evaluation datasets is facilitated by QCTool, an annotation tool written in Python and based on XTrans. QCTool allows annotators to view source documents and translations side-by-side and edit the working copy of a translation. It also includes functionality for viewing and reverting to previous translation versions, flagging sections for further review, playing back source audio data, and displaying edits as they are made.

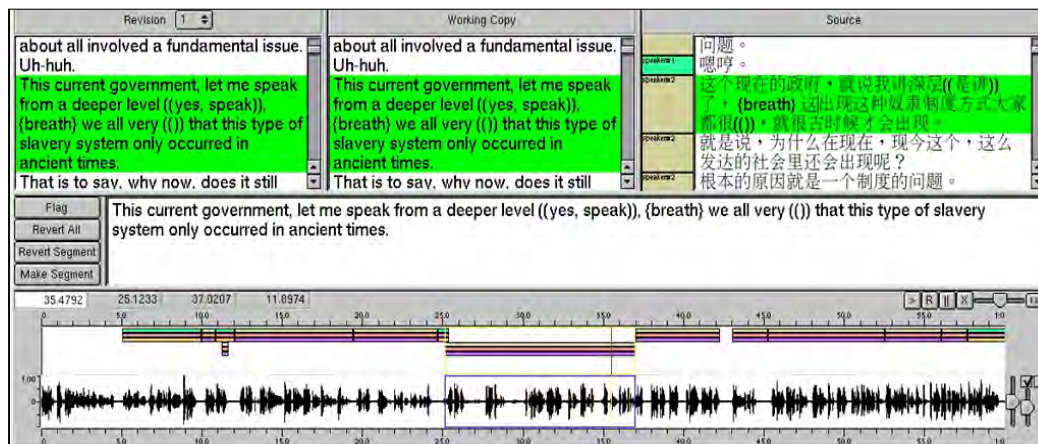


Figure 1.5: Screenshot of QCTool.

QCTool was developed by reusing and specializing components of XTrans. A simple analysis shows that QCTool reused 99% of the XTrans code. About 8% of the QCTool code was newly added, due to the object-oriented design of XTrans. For example, the text display required enhancement to ensure that corresponding segments are aligned when displayed on the tree panel text display. It is this approach that made the rapid development of the tool possible.

As with LDC's other annotation tools, QCTool is fully integrated with LDC's Annotation Workflow System (AWS), which manages documents, directories, permissions, and assignments, and tracks annotator efficiency and progress. Together, QCTool and AWS ensure that each intermediary version of the translation is stored for later training and analysis. In LDC's production pipeline, QCTool is integrated with AWS for Steps 3, 4 and 5 of the 6-step process.

Before entering the AWS workflow, the file contains two layers of text: source text and a preliminary translation. At each step of the workflow, AWS adds an exact copy of the previous translation layer. The annotator at each step works on the most recently added layer, which is displayed in the "Working Copy" panel, to correct and improve that layer. One of the previous translations is displayed on the "Revision" panel, and the user can select which revision to display. The source text is displayed on the "Source" panel.

QCTool uses a slightly modified version of the original data model used by XTrans. The original data model is essentially a table where each row represents a segment or sentence. The translation QC process involves several layers of such tables. For example, source text, preliminary translation and corrections from additional QC steps each form a

layer. In QCTool, these layers are combined into one by means of a table ID; that is, each row is augmented by an ID of the table it belongs to. The new data model is physically stored using the same file format used by XTrans. However, only QCTool recognizes the additional information to display each row in an appropriate text panel.

QCTool increases the speed and accuracy of the quality control process by giving the annotator access to all of the relevant information in one place, and making it much easier to visualize revisions and identify problematic regions.

Translation Pipeline Database

LDC's core translation infrastructure is grounded in a custom MySQL database that tracks every file at every stage of the translation pipeline and allows the steps outlined above to be followed smoothly, efficiently, and consistently.

This database stores information on data features (e.g., token count, language, genre) and partitions (devtest, training, evaluation); tracks assignments, deadlines, and payments for outsourced translation; and captures information on data location and storage that is exploited by downstream processing scripts. The database is also the backend for the Translation Extranet, which allows files to be stored securely but easily accessed and managed by external agencies and LDC administrators via a simple web interface.

This database is the key to data management of this scale and complexity. Fields are generally populated via batch imports, and files are added at the planning stages so that each step in the pipeline can be tracked and recorded.

Querying allows programmers and managers to drill down and retrieve all of the relevant information associated with a particular file (e.g., which kit it belongs to, when it was delivered, what the QC score was, which agency translated it, whether it has been processed, etc.), as well as, all the files associated with a particular stage (e.g., processed but not released), feature set (e.g., Arabic Broadcast News from Phase 2 Training), source (e.g., Xinhua), translation agency, and so on. Processing scripts hook into the database to improve efficiency and allow certain fields to be automatically updated in real time. Additionally, reports synthesizing various points of interest can be generated on the fly for better planning, management, and information-sharing.

Transcription and Translation File Format

ID	Label	Description	Format
1	file	file name or id	Unicode
2	channel	audio channel	Int
3	start	start time	Float
4	end	end time	Float
5	speaker	speaker name or id	Unicode
6	speakerType	speaker type	Unicode
7	speakerDialect	speaker dialect	Unicode
8	transcript	transcript	Unicode
9	section	section id	Int
10	turn	turn id	Int
11	segment	segment id	Int
12	sectionType	section type	Unicode
13	suType	SU type	Unicode

Table 1.4: The 13 fields of a record, or segment, in XTrans .tdf format.

The output of XTrans and QCTool is a simple Tab Delimited Format (TDF), which is the primary file format for transcripts and translations produced at LDC. This file format represents data as a set of “records”, which are in turn a set of “fields” separated by tab characters. Each record refers to a particular segment in the transcript.

In addition to the body of segments, there are several lines of meta-information in the .tdf file. The first line declares the above field specification for segments in the following form:

```
file;unicode channel;int start;float ...
```

The second and third lines specify where the location and types of the section boundaries. For example,

```
;;MM sectionTypes [u'report', u'nontrans', None]
;;MM sectionBoundaries [0.0, 425.3, 9999999.0]
```

These lines mean that the first section starts at 0.0 second and its type is “report”, and that the second section starts at 425.3 seconds and its type is “nontrans”, and this is the last section (9999999.0 is always the last field in this line).

1.3.1.5. Pilot Annotation

LDC transcription and translation efforts aim to maximize the utility of data for GALE teams; this vision includes producing pilot annotations and experimenting with new approaches to provide researchers with additional resources to support system development and testing.

Supralelexical Annotation

In GALE Phase 3, LDC performed pilot annotation to enrich devtest and unsequenced evaluation transcripts from previous GALE phases. The supralelexical annotation pilot aimed to maximize usefulness of existing transcripts for research teams, and involved in-line transcript annotation and named-entity annotation. The supralelexical annotation task was performed in two stages: the first stage required annotators to listen to the audio file while annotating the transcript and included the following tasks:

- marking filled pauses
- confirming accuracy of speaker info (gender & native/non-native status)
- adding in-line notation about bandwidth, which included surrounding transcribed words that sounded like telephone speech with <telephone> </telephone> tags, and digitally-transferred field reports with <field report> </field report> tags
- inserting speaker-created and background noises
- adding Arabic dialect descriptions for non-MSA utterances

Since Arabic dialect annotation is a specialist task, supralelexical annotators only changed non-MSA tags when they could confidently identify the dialect. Annotators chose one of the following dialect descriptions: (1) Maghrebi, (2) Egyptian/Sudanese, (3)

Levantine, or (4) Gulf/Iraqi. When annotators could not confidently identify dialect being spoken, they retained the non-MSA tags from the original transcript. Any region not tagged with a dialect ID or as non-MSA was assumed to be MSA (LDC 2009).

The second stage of supralephical annotation involved marking named entities as Person, Title, Organization, or Location (LDC 2006).

Language	# files BC	# files BN	Approximate total hours
Arabic	102	152	8.5
Chinese	157	144	10

Table 1.5: Data volumes annotated for the Phase 3 supralephical annotation pilot.

Translation Pilot Experiments

LDC’s streamlined translation processes have led to a previously impossible level of adaptability to new techniques or datasets and allow quick responses to innovative data ideas. For example, LDC performed a pilot experiment called “MT Plus” wherein source data was run through in-house MT systems and then sent out to translation agencies for correction. GALE sites were interested in comparing MT-based translation to standard from-scratch translation.

Pilot experiments have also been undertaken in translation from English into Arabic/Chinese, in first-pass translation without quality control, and in sentence-level translation. LDC’s efficient infrastructure means that resources are maximized and start-up costs are minimized.

1.3.1.6. Conclusion

This section described LDC’s efforts to produce training and evaluation data, including transcripts and translations, to support the DARPA GALE program. GALE’s challenging performance targets demand linguistic resources on a scale and complexity never before encountered. LDC has developed a robust and flexible pipeline, combining enabling technical infrastructure, detailed task specifications and fully documented best practices. Over the life of the program resource creation at LDC has become more efficient and adaptive, with increased emphasis on automation and utilization of emergent GALE technology to improve and augment the data pipeline. These combined efforts have enabled LDC to meet and often exceed requirements for large volumes of translation and transcription training data, high quality evaluation and development sets, and informative annotation experiments that are responsive to the ever-evolving needs of system developers, evaluators and program sponsors.

1.3.2. Word Alignment for Improved Machine Translation

Authors: Xuansong Li, Xiaoyi Ma, Stephen Grimes, Stephanie Strassel, Gary Krug, and Dalal Zakhary

1.3.2.1. Introduction

Manual word alignment annotation at LDC is a part of the GALE project and is an ongoing five year effort. This project has produced Arabic-English and Chinese-English word alignment corpora with the amounts of data shown in Table 1.6.

	Genre	Phase 1	Phase 3	Phase 4	Total
Arabic	BN	25	79	89	193
	BC	31	105	n/a	136
	NW	105	n/a	266	371
	WEB	94	26	85	205
Chinese	BN	35	95	n/a	130
	BC	33	75	57	165
	NW	102	n/a	149	251
	WEB	102	33	126	261

Table 1.6: Data volume in thousands.

1.3.2.2. Data and Tool

Data type and source

Two types of source languages involved in the current task are Arabic and Chinese. In the first year of the GALE project, the data types used included newswire, broadcast news, broadcast conversation (talk shows, call-in shows), newsgroups and weblogs. Newswire text came from tree-bank data while other types were harvested at LDC and translated by different translation agencies. In Phase Three of GALE, data types were reduced to broadcast news, broadcast conversation, and web (including newsgroup and weblogs). These data are now or will be tree-banked. The data source is shown in Table 1.7 Table 1.8.

Genre	Chinese Data Source
BN/BC	2005-2006 CCTV, 2005 Phoenix TV, 2006 CCTVNEWS, CCTV4, HUBEI 2007 Phoenix TV, CCTV, CCTVNEWS, China Central TV 2005-2006, CCTV, 2005 Phoenix TV
NW	1994-1998 Xinhua
WEB	Online, VOA

Table 1.7: Chinese data source.

Genre	Arabic Data Source
BN/BC	2005 Aljazeera, 2001 Nile TV, 2005-2006 Al Arabiyah, 2007 Saudi Nightly News, Dubai TV, Al Iraqiyah, Kuwait TV News, Lebanese Broadcast Corp, Saudi Nightly News 2005 Aljazeera, 2001 Nile TV
NW	2002 An Nahar
WEB	Online, VOA

Table 1.8: Arabic data source.

The data was selected from the released source of LDC GALE Phases 2 and 3.

Tokenization

For a more precise text and punctuation alignment, both the source and translation were tokenized automatically, without human interventions. The tokenization of English follows the same guidelines used in Penn English Treebank: split words by white spaces, split clitics/contractions (such as: shouldn't, I'm, etc.), separate punctuations from the preceding/following words. Apostrophe S ('s) is treated as a separate token. Penn English Treebank treats most hyphens as separate tokens; however, some are treated as part of words. Arabic tokenization follows the ATB tokenization scheme by directly extracting tree tokens from ATB for word alignment tasks.

Because of a lack of word boundary, Chinese tokenization is challenging due to segmentation issues. We finally chose character-based files for tokenization with the following considerations. First, word segmented files are normally automatically done by a word segmentation tool and a predefined monolingual dictionary. Yielded errors need to be manually corrected, which is another added process to word alignment itself. Second, character-based system can be easily adapted to any existing segmentation system. Third, character-based files provide the flexibility of ignoring a word boundary when necessary and breaking down a two-character word to minimum semantic units, which greatly benefits cross-cultural findings and more subtle linguistic descriptions. Fourth, the convenience of handling new words is obvious with character-based files, such as proper nouns and newly coined words. These types of words usually require a great effort in manual correction after the texts have been word-segmented. Finally, Chinese input texts at character level can simplify data pre-processing. Therefore, in tokenization we treat each Chinese character as a separate token. All hyphens are separate tokens. In addition, we separate punctuations from the preceding/following characters.

Data structure and format

The input data, including both source and translation data, are in tab delimited format (.tdf). The annotation results are stored in an xml-like format, as illustrated in Figure 1.6.

The key tags are:

Source: the source sentence (in Arabic or Chinese).

Translation: the English translation.

Matrix: a two dimensional array \mathbf{M} indicating alignment among words. The rows contain the indices to the English tokens and the columns, the into the source (Arabic/Chinese) sentence tokens. The value of each cell can be 0, 1 or 2.

$\mathbf{M}[i, 0]$ represents the “Not Translated” English tokens, $\mathbf{M}[0, j]$ represents the “Not Translated” source tokens. A value of 1 indicates “Not Translated” and “Correct,” meaning a word alignment cannot be found, but the meaning is conveyed in the translation. A value of 2 indicates “Not Translated” and “Incorrect,” meaning a word alignment cannot be found, and the translator has made an error (missing or inserted word), which is rare. For example, if $\mathbf{M}[3,0]$ is 1, it indicates that the third English token is not translated, but the translation is correct.

If $M[i, j]$ is 1 or 2, it indicates there is a link between English token i and source token j . A value of 1 indicates the translation is correct, and value of 2 indicates incorrect, which is rare. There are cases where more than one English token are aligned to more than one source token, in which case, links will be drawn between every English token and every source token. For example, if English tokens i and $i + k$ align to source token j and $j + l$, $M[i, j]$, $M[i + k, j]$, $M[i, j + l]$ and $M[i + k, j + l]$ will be set to 1 or 2, depending on whether the translation is correct.

“Unaligned_sentence” between matrix tags indicates the sentences rejected by annotators during annotation in cases where sentences are half translated, foreign text, stylistic text of peculiar data types, or locally ill-formatted. For instance, in broadcast conversations, there are fragments of music which are represented as blank lines, and in newsgroups and weblogs, there is foreign text, email address or web links. All data are encoded in UTF8.

```

<seg id=1>
<source_raw>新华社拉萨二月二日电(记者央珍) </source_raw>
<source>新华社拉萨二月二日电(记者央珍)</source>
<translation_raw>Xinhua news Agency, Lhasa, February 2, (reporter Zhen Yang)</translation_raw>
<translation>Xinhua news Agency, Lhasa, February 2, (reporter Zhen Yang)</translation>
<matrix>
00000000011000000
01110000000000000
01110000000000000
01110000000000000
10000000000000000
00001100000000000
10000000000000000
00000110000000000
00000001000000000
10000000000000000
00000000001000000
00000000000110000
00000000000001100
00000000000000110
00000000000000011
</matrix>
<time>100</time>
<comments> </comments>
</seg>
<seg id=2>

```

Figure 1.6: Data File Format.

Tool

The alignment tool is a simplified version of a JAVA-based word alignment tool developed by Carol Nichols (c1n23+@pitt.edu) and Rebecca Hwa (hwa@cs.pitt.edu) at the University of Pittsburgh (Nichols 2005). The original tool has a client-server model and works for Chinese-English only, and the server side is very

CPU intensive. To better serve our task, the server side is discarded and only client side (GUI interface) is kept. Instead of JAVA applet, the tool has been changed to work as a stand-alone GUI application, and Arabic support has been added. The GUI interface has been improved for both visualization and functionality of annotation. The tool presents each pair of tokenized sentences vertically, which are clickable for one-to-one, one-to-many, many-to-one, and many-to-many links. To handle noisy data or style features of different data types which are not appropriate for alignment, a “reject-segment” button was created to conveniently exclude the inappropriate data from alignment while maintaining the integrity of source data.

1.3.2.3. Annotation Guidelines

Guideline compilation process

Guidelines compilation began with the annotation of a pilot set of files by native Arabic and Chinese lead annotators. The first guidelines draft was written based on initial annotation results, with reference to the Annotation Style Guide for the Blinker Project (Melamed 1998). After several rounds of discussions, the draft was revised to form the first version. Two Chinese annotators and two Arabic annotators then joined in and annotated the trial sets according to these guidelines. Their annotation results were examined carefully for differences, which were discussed and tackled again in revising the guidelines. These guidelines were consistently updated to tackle new findings. During the annotation process in Phase1 and Phase 3, guidelines were updated several times due to newly found untackled language phenomena or direct requirements from the GALE research sites. The basic approaches of guidelines, however, remain the same.

Guideline approaches

Two versions of guidelines have been developed: Chinese-English and Arabic-English. The same framework is adopted for the two types of guidelines, with the general principles and strategies discussed first, then the specific rules and language issues. Ample examples are provided to support the annotation rules. The most recently updated word alignment guidelines `GALE_Arabic_alignment_guidelines_v3.0.pdf` and `GALE_Chinese_alignment_guidelines_v3.0.pdf` can be accessed and downloaded from <http://projects ldc.upenn.edu/gale>.

Word alignment is a task performed between parallel texts, involving two languages and addressing universal linguistic phenomena as well as cross-cultural features. To respond to universal linguistic issues, general principles are discussed, while specific rules are stipulated to tackle cross-culturally different topics.

Language universals

a. Alignment types:

Two types of links (translated-correct and translated-incorrect) and two types of marks (“not-translated correct” and “not-translated incorrect”) are established to capture linguistic information and cross-cultural elements.

b. Translated:

Pure linguistic translation equivalence refers to semantic “deep structure” equivalence either with or without lexical/functional surface-structure equivalence. A word can be translated in various ways. As long as the meaning is adequately expressed, the word is semantically “translated” in spite of variations in surface-structure. Passive voice in one language can be translated into active voice in another, or there may be divergences in diction, as in:

高兴(happy)死(die)了。
Extremely happy.

All such variant versions are semantic translations of the source, and thus “correct” links. This is true for all languages. Aligning correct links is important to word alignment task because valid translation pairs can be conveniently extracted to construct translation lexicons. Most of the links in our task are of this kind. We designed “Translated incorrect” link type to cover instances where the text is improperly translated in meaning or grammatically wrong.

c. Not-translated:

If a word is not translated or conveyed, thus causing information loss, then it is regarded as “not translated”, and naturally is “incorrect.” Therefore, “not-translated incorrect” refers to cases with a loss of deep structure meaning and an absence of surface structure representation, as in the following example, “butter,” is missing in deep structure as well as surface structure in translation.

他(he)买(buy)了牛奶(milk)黄油(butter)和(and)鸡蛋(egg)。
إشترى الحليب والزبدة والبيض.
He bought milk and eggs.

Translating cross-cultural thought inevitably involves translation variations, which are realized by means of expansion, deletion, summarization, explanation, supplement, combination, and reformation in the light of the special needs of readers. Such adaptations and translation variations lead to superficial or surface structure changes, such as overt additions or omissions. These words are extraneous. Deleting them, however, would corrupt the correctness of the sentence, as with determiners which are absent in Chinese but present in English. The additions or omissions are, from a cross-cultural perspective, obligatory and systematic. Special grammatical rules in different languages can be revealed by studying omissions or additions in translation. To this end, the “not-translated correct” markup is designed to label these extras in source/translation.

Our primary concern is how to adequately categorize these superficially extraneous words or phrases. They can be divided into two types: functional (grammatical) and semantic. Furthermore, they can be either at word-level, sentence-level, or discourse level. In English translations, the functional (synsemantic) words, such as prepositions, determiners, subordinating conjunctions, certain particles, auxiliary, and modal verbs are mostly word-level or local level tokens and cannot be neglected. They are so rudimental

in revealing the hidden information regarding constituent dependencies that leaving them unaligned would lose valuable dependency information. Moreover, word-level synsemantic words contribute to revealing the complete semantic equivalence between languages, without which the language would be unacceptable. Therefore, synsemantic words at the word level are attached to their autosemantic head words and they jointly constitute “translated correct” links. For instance, “the” can be attached to “reporter,” and “the reporter → 记者 (reporter)” is an aligned “translated correct” link. Describing word relations at a local level by the attachment approach is import for word alignment tasks because word relations can help to capture local level syntactic information.

Extraneous words at sentence level usually carry strong cross-cultural or discourse significance. They are not important for semantic equivalence. Without them, the translations are still correct or acceptable. Therefore, these added words with no direct lexical equivalence can be marked as “not translated correct.” They are not translated in the sense that they have no written equivalents. However, they are correct in the sense that they are needed for being more like English or Chinese, or they are grammatically needed. Language idiosyncrasies play an important role here. Some words appear or disappear depending on the peculiarities of a specific language. For instance, “都 (all)” in the following example does not carry meaning, but it makes the Chinese sentence more fluent. Marking these words is important to machine translation in the sense that a grammatical/rule-based translation model can choose to ignore such words.

国家 (nation) 每 (every) 年 (year) 都 (all) 拨 (allocate) 专
(special) 款 (fund) 用 (use) 于 (for) 开展 (develop) 残疾人
(handicapped) 体育 (sports) 活动 (activities).
Every year the nation allocates special funds to be
used in developing handicapped sports activities.

Currently, pragmatic and contextual features are also covered under “not-translated correct” category. Discourse-level extraneous words rely on discourse clues and no direct lexical correspondence could be found locally within a phrase or sentence. For these cases, “not-translated correct” is the best solution. For instance, in the following example, the word 中国 (China) is omitted from translation and can be labeled as “not-translated correct.” Here, 中国 (Chinese) 排球 (volleyball) 队 (team) is semantically equivalent to “the volleyball team” if we carry on the same topic. However, if they are examined at the word-level or locally, we can never say that these two terms are semantically equivalent. They are equivalent only by adding discourse clues.

中国 (Chinese) 排球 (volleyball) 队 (team) 昨日 (yesterday) 抵达 (arrive) 美国 (the
U.S.). 中国 (Chinese) 排球 (volleyball) 队 (team) 今天 (today) 下午 (afternoon) 与
(and) 当地 (local) 纽约 (New York) 队 (team) 进行 (have) 了 比赛 (match).
Chinese volleyball team arrived in the U.S. yesterday.
The volleyball team had a game with the local New York
team this afternoon.

Likewise, insertions or omissions of personal or genre stylistic features may not be translated, and no pair equivalency can be detected. They are not very important semantically, thus are treated as “not-translated correct” if no match is found.

d. Minimum Match:

Another alignment issue concerns the recognition of link boundaries. The “minimum match” approach is proposed for finding complete and minimal semantic translation units. In word alignment, the principle of a word-for-word link is strictly followed, that is, the smallest number of words will be preferred, such as the case “鲜(fresh)” in “鲜(fresh)花(flower)” is paired to “fresh” in “fresh flowers” instead of treating “鲜(fresh)花(flower)” as a whole correspondent to “fresh flowers.” This minimum unit approach is important to word alignment because without these minimum unit alignments, minimum syntactic structures units built on these units then cannot be adequately established.

However, some links are many-to-many links since they cannot be separated from each other. Such cases include abbreviations like “世界(world) 卫生(health) 组织(organization)→WTO,” idiomatic expressions/transliteration phrases like “齐(in line with) 头(head) 并(and) 进(march)→keep abreast with,” “heart is racing→热(hot) 血(blood) 沸腾(boil),” “in your dreams → في المشمش,” “the youngest child→ آخر العنقود,” and other inseparable cases. Hyphenated words in English are treated as one unit if the subparts of hyphenated words are inseparable. However, if salient one-to-one links exist on both sides, they are paired separately, especially with nonce words.

e. Language peculiarities:

While a hidden matching semantic structure of two languages with cross-cultural features can be revealed by systematically reoccurring omissions/insertions, the study of peculiarities of one language can indicate idiosyncrasies with no systematic matching structure in other languages. A large part of the guidelines is reserved for these language idiosyncrasies with specific rules, which have been augmented with new findings of language peculiarities during annotation process. Arabic guidelines deal with Arabic specific features, including equational sentences, empty subject, cliticization of determiners, prepositions, pronouns, and conjunctions, as well as idioms and particular Arabic interrogative words with no equivalent in English. In Chinese-English alignments, featured idiosyncratic topics include non-inflection, topicalization, measure words, duplication, tense and aspects, various types of helping words, etc. Chinese 的, sometimes described as “evil” or “notorious” due to its uncertain features, requires several annotation rules for a comprehensive description. A glimpse of the following examples can well tell its multi-function and extreme uncertainty idiosyncrasies: adjective modifier as in a); adverb modifier as in b); possession modifier as in c); noun modifier as in d); location (general) modifier as in e); clause modifier as in f); tense modifier as in g). During annotation, 的 is either aligned to its counterparts as in c), d), and f) or attached to related words as in a), b), e), and g).

a) 红红(red) 的→red (的 attached to 红红)

b) 高兴(happy)地→ happily (地 attached to 高兴)

- c) 中国(China) 的产品(product) → China's product (的 aligned to ``s")
- d) 人生(life) 的历程(cause) (→ the cause of life (的 aligned to "of")
- e) 桌子(table)上(on)的书(book) → book on the table (的 attached to 上)
- f) 经历(experience)过(past)战争(war)的(DE)人(person) → those who have experienced wars (的 aligned to "who")
- g) 呈交(submit)的报告(report) → report submitted (的 attached to 呈交)

1.3.2.4. Evaluation

To test the inter annotation rate, dual annotation is done using AWS workflow system, which blindly assigns a number of files to two annotators at the same time. The results are then computed to show the rate of difference. Because of translation quality and genre of files, the agreement rate for newswire files is obviously higher, as shown in Table 1.9, whereas the rate for web and broadcast news is relatively lower, with a range of 80%-90%. Table 1.9 shows the inter-annotator agreement of two annotators on 4 newswire files (F1, F2, F3, and F4). The first row of the table shows the total alignment links from each annotator. The agreement is computed based on their common links. When re-assigning the same file to the same annotator after an interval of about one month, the intra annotation rate shows a good result of 95%-99% for the newswire data.

	Pass	F1 links 108-108	F2 links 190-185	F3 links 273-269	F4 links 215-218
Precision	1P	87%	85%	82%	85%
Recall	1P	87%	88%	83%	84%
F-score	1P	87%	86%	83%	85%
	Pass	F1 links 106-105	F2 links 182-182	F3 links 262-257	F4 links 220-216
Precision	2P	95%	95%	91%	90%
Recall	2P	96%	95%	93%	91%
F-score	2P	96%	95%	92%	91%

Table 1.9: Inter-annotator Agreement on 4 Newswire Files.

1.3.2.5. Future Work

Word alignment is a complex task requiring an exhaustive description of translation unit equivalence between source and target languages. The work reported in this section established a framework for alignment. With IBM we then jointly designed and deployed an enhanced alignment and tagging annotation scheme as part of the Phase 4 Chinese-

English alignment task. As a next step, this tagging scheme will be implemented for other language pairs including Arabic-English.

Another avenue for further research is the inclusion of additional levels of alignment. Initial investigations during Phase 4 on automatic post-processing to induce higher level alignment proved fruitful; this has yielded a promising automatic annotation structure for creating multi-level alignments during Phase 5. The focus of Phase 5 word alignment will be on infrastructure and technologies for streamlining the creation of multi-level parallel aligned Treebank corpora.

1.3.3. Distillation Training Data

Authors: Heather Simpson, Stephanie M. Strassel, Zhiyi Song, Robert Parker, Kazuaki Maeda

1.3.3.1. Introduction

Distillation is the final stage of language processing in the GALE program, in which relevant information is extracted from foreign language and English input and concisely presented to users in English. It is not a key-word search, and does not involve summarization, but rather it utilizes language analysis techniques to identify information relevant to a user's query, with the aim of extracting all available relevant information and eliminating redundancy in that information. The GALE Distillation evaluation is designed to quantify the amount of relevant and non-redundant information a distillation engine is able to produce in response to a given template query. Linguistic Data Consortium creates and distributes linguistic resources for distillation system training, including source data, annotations, inter-annotator agreement studies, annotation GUIs and software, and annotation guidelines and related documentation.

1.3.3.2. Distillation Source Data

Creation of training data for distillation begins with selection of source documents that provide the basis for human annotation. Distillation source data is derived from existing LDC corpora, combined with new material collected by LDC specifically for GALE. To support a variety of approaches to distillation system development, LDC targets source data in all three GALE languages (Arabic, Chinese and English) and all four genres (newswire, web text, and transcripts of broadcast news and broadcast conversation). Selection of source material for each phase is informed by research sites' needs, and is further guided by the particular evaluation requirements for each phase. In general an effort is made to select documents that are likely to be most useful for system training; for instance, documents that are rich in named entities and documents known to be relevant to training data queries are favored over more general news documents. After semi-automated selection, source documents undergo data formatting and processing as needed to conform to the GALE program standard. Source data may be automatically or manually segmented into sentences prior to distribution and annotation, and documents are indexed for use with Querier, LDC's internally developed search engine. After indexing, the source data enters the annotation pipeline. Note that not all selected source data is

manually annotated, because not all documents prove relevant to the targeted training queries, but at this stage all selected data is available for annotation.

In addition to training source data, LDC also selects, formats and distributes test data to support the annual evaluation of Distillation engines. Evaluation source data requirements are specified by BAE Systems who is responsible for conducting the performance evaluations and for providing the gold standard evaluation references. Evaluation source data may be identical to training data (but with different queries); or it may be distinct, depending on the current evaluation protocol. Evaluation data may also require additional work by LDC prior to its distribution to BAE; for instance, in some cases LDC provides English (human) translations of foreign language source text to enable analysis of those texts by English-speaking judges at BAE.

1.3.3.3. Annotation Approaches

1.3.3.3.1. Query Annotation

Distillation annotation begins with queries. At the start of each GALE phase, BAE Systems develops a set of query templates that are appropriate to that year's evaluation focus. Query templates consist of argument variables for persons, organizations, dates, locations, events, topics and so on; for instance “*WHERE HAS [person] BEEN AND WHEN?*”. To satisfy training data requirements it is necessary to develop a large set of specific training queries that conform to the available templates for each new phase of the program, with argument variables replaced with specific entities and events. To ensure consistency between evaluation and training data development approaches, BAE first develops a dozen or more exemplar queries in English, then LDC continues to develop additional queries as needed.

For the initial phases of Distillation, LDC created queries based on a search of external news sources. This was an efficient method for query creation, but resulted in some queries having little or no representation of answers in the corpus. In later phases LDC improved upon the process by requiring annotators to search the training source data corpus to generate new queries. Searching the source data for entities and events with attested query answers is significantly more time-consuming, but results in a set of queries with richer representation of answers. Query creation is facilitated by a simple GUI that allows senior annotators to manually input queries intended for downstream annotation into a form that output queries in the required query XML format. Because annotation of each query may take place across languages, initial English queries are also translated by LDC into Chinese and Arabic.

Once queries are in place, manual annotation begins. Native Arabic, Chinese or English annotators submit their assigned query into the search engine component of the annotation toolkit. The tool returns a relevance ranked list of documents, which annotators briefly review and label as relevant or off-topic. As soon as a document is labeled YES, the annotator then extracts relevant snippets that are subject to further annotation. A snippet is defined as a continuous string of text (noun phrase, clause and/or sentence) that contains an answer to the assigned query. Annotators follow template-specific relevance guidelines provided by BAE when determining whether a string of text is relevant to a given query.

Because snippets are extracted directly from a document, they often contain pronouns, temporals and locative constructions whose reference is ambiguous within the selected string. In such cases, annotators must attempt to disambiguate the pronoun or other word based on information provided elsewhere in the document, as in the following example where *here* can be resolved to a specific location based on document context:

Iraq's Deputy Prime Minister Tariq Aziz was to meet with Russian Foreign Minister Andrei Kozyrev here [Moscow] on Wednesday.

For some phases of Distillation, snippet extraction and antecedent resolution is followed by manual nuggetization. A nugget is a fact or a statement extracted from a snippet. Nuggets describe any piece of information that an annotator considers a valid answer to the query. If a snippet contains a fact that is unrelated to the query by any standard of relevance, no nugget is created for that fact. Any piece of information that might have some relevance to the query is nuggetized. For instance, the following sentence has three nuggets.³

Pinochet, who is 82 and who stepped down as commander in chief of the Chilean military, was arrested by the British police.

Nugget 1: Pinochet is 82

Nugget 2: Pinochet stepped down as commander in chief of the Chilean military

Nugget 3: Pinochet was arrested by the British police

Nuggets might provide a description of a named entity or assert that a certain event took place. They may also provide temporal, locative, causal, or other types of information that further describe an event. Detailed annotation guidelines created by LDC instruct annotators in the procedural approach to nuggetization.⁴

Some phases of Distillation also incorporate nugget co-reference, both within and across languages. Semantically equivalent nuggets for a single query and a single language are first grouped into “nugs” by senior annotators. Then, after all nugs for a given query have been established for each language independently, one group of bilingual lead annotators co-reference English and Arabic nugs, while a separate team of bilingual lead annotators co-reference Chinese and English nugs. Arabic and Chinese nugs co-referenced to the same English nug are automatically merged to create a trilingual “supernug”. Arabic and Chinese nugs that remain unmapped to any English nug at this stage are further reviewed, and English translations are created to enable further co-reference between Chinese and Arabic nugs. This additional co-reference was provided in order to improve systems’ performance in identification of redundant information. The resulting list of supernugs for each query represents a complete list of “facts” in English, drawn from all of the multilingual, multi-genre documents that were judged as relevant to the query.

³ Possible nuggets are determined not only by the sentence content but also the constraints of the specific query.

⁴ See <http://projects ldc.upenn.edu/gale/Distillation/> for links to annotation guidelines.

The nugget co-reference task proved to be quite difficult and time-consuming. Annotators struggled to agree whether the pairs of nuggets should be co-referenced, since many nuggets contain related but not completely identical information as in the following examples.

Nugget 1: 14 suicide bombers killed 28 weekend revelers

Nugget 2: 28 civilians were killed

Nugget 1: Tariq Aziz was a minister in Saddam's government

Nugget 2: Tariq Aziz was an Iraqi official in the Saddam regime

Although LDC's annotation guidelines provided detailed instructions and examples for resolving such uncertainty, the task remained challenging and time-consuming. Later phases of Distillation drop manual nuggetization and nugget co-reference altogether, in response to BAE adopting automatic nuggetization as part of the evaluation protocol.

1.3.3.3.2. Entailment Annotation

To support Distillation's goal of identifying redundant and contradictory information, and in lieu of nugget co-reference annotation which proved difficult for annotators to do consistently, LDC developed a supplemental annotation task in Phase 2. In Entailment Judgment annotation, ordered pairs of nuggets for each query were presented to annotators and labeled as follows:

- **Redundant:** Nugget B is redundant with respect to nugget A if A entails B and if nuggets A and B refer to the same event or fact.
- **Equivalent:** Nugget A entails nugget B and B entails A, and nuggets A and B refer to the same event or fact.
- **Contradictory:** A and B contradict if assuming that A is true, it can never be the case that B is also true.
- **Not related:** Nuggets A and B stand in an entailment relation but do not refer to the same event or fact
- **Not Sure:** Annotator is not sure; especially used when it cannot be determined whether nuggets refer to the same fact or event.

Portions of the data were also dually annotated for the purpose of assessing annotator consistency. Entailment annotation proved to be an efficient annotation task when compared to nugget co-reference, with significantly higher annotation consistency.

1.3.3.3.3 System Output Relevance Assessment

One additional annotation task introduced in later phases of Distillation is assessment of system-extracted snippets. Systems process LDC-supplied training queries, and return extracted snippets for LDC assessment. All extracted snippets are in English, either from English source documents or foreign language snippets translated by GALE systems into English. No limits are placed on the number of snippets extracted per query. Annotators

judge each system snippet as *Relevant* or *Irrelevant*, following the existing relevance guidelines. In later phases these categories were expanded to include *Partially Relevant*, *Too Little Information* and *I Don't Know*. In some cases snippet relevance is impossible to determine, either because of poor translation quality or lack of context. Such snippets are judged as *Unknown*.

1.3.3.4. Challenges and Conclusions

To respond to evolving research approaches and evaluation protocols, LDC's approach to training data creation was slightly different for each phase. This presents a number of challenges, most notably the need to define new annotation guidelines and adapt annotation infrastructure like GUIs and data processing scripts. The most notable changes over time were the addition of new templates, the changing approach to manual nuggetization and nugget co-reference, and the introduction of the system assessment task. During these task changes a primary concern was maintaining annotation consistency, both within LDC and between LDC and BAE. Although the process used by BAE to develop gold standard evaluation data necessarily differs from LDC's approach to training data creation, BAE and LDC relied on the same annotation guidelines for templates, judging relevance, and defining nuggets. Particularly in early phases of GALE annotation managers from each organization met regularly to discuss approaches and review consistency. Despite these mutual efforts some difficulties persist. In particular, relevance guidelines proved to be underspecified when confronted with real-world data. Guidelines for some templates, particularly unstructured and open ended ones, leave significant room for subjective interpretation by the annotator. In such cases LDC introduced and documented rules of thumb for the annotation team to follow. Still, studies on inter-annotator consistency for relevance judgment on system-extracted (and machine translated) snippets ranges from 59-89%, depending on template and annotator (c.f. Babko-Malaya, 2008).

Across the four phases of Distillation, LDC has distributed source data, transcripts and translations plus relevant snippets, with optional nuggetization, for 1148 English, Chinese and Arabic queries. Additionally, LDC has distributed entailment judgments for 60 English queries and has assessed relevance of system-extracted snippets for several hundred queries.

1.3.4. Arabic Treebanking

Authors: Mohamed Maamouri, Ann Bies, Seth Kulick, Nizar Habash, Reem Faraj and Ryan Roth

1.3.4.1. Introduction

Collections of manually checked syntactic analyses of sentences, or *treebanks*, are an important resource for building statistical parses and evaluating parsers in general. Rich treebank annotations have also been used for a variety of applications such as tokenization, diacritization, POS tagging, morphological disambiguation, base phrase chunking and semantic role labeling.

Two prominent efforts in Arabic Treebanking existed prior to GALE: the Penn Arabic Treebank (PATB) and the Prague Arabic Dependency Treebank (PADT). The main difference between these two resources is the linguistic representation: PATB uses phrase structure and PADT uses dependency representation. Both of these important efforts employ complex and very rich linguistic representations that require extensive annotator training. More concretely, the PATB provides tokenization, complex POS tags and syntactic structure, in addition to diacritizations, lemma choices and some semantic tags (such as TMP and LOC) to distinguish different modifiers. PATB annotation consists of two phases: (a) Morphological/Part-of-Speech (=POS) tagging which divides the text into lexical tokens and includes morphological, morphosyntactic and gloss information, and (b) Syntactic analysis referred to as Arabic Treebanking (=Arabic TB) which characterizes the constituent structures of word sequences, provides function categories for each non-terminal node, and identifies null elements, co-reference, traces, etc. (similar to the Penn English Treebank II style) (Marcus *et al.* 1994; Marcus *et al.* 1993; Bies *et al.* 1995). In addition to the usual issues involved with the complex annotation of data, the work on Arabic treebanking, spearheaded by the LDC's PATB effort since 2001, has caused the field to come to terms with a number of issues that are specific to a highly inflected language with a rich history of traditional grammar (Maamouri and Bies 2004) in a variety of ways.

Within GALE, the work on Arabic treebanking was manifested in two efforts. First was the continuing annotation and upgrading of the PATB with a focus on unifying the different PATB parts. Second was a new exploratory annotation effort called the Columbia Arabic Tree Bank (CATiB). CATiB contrasts with PATB (and PADT) in putting an emphasis on faster production with some constraints on linguistic richness. Like PATB, CATiB annotation also consists of two levels of annotation: POS and syntactic. However, because of the different choices made in the CATiB annotation style, these two levels contain different amounts of information. Like PADT, CATiB uses a dependency representation, however with much less morphological information and shallower syntactic relations.

The intent behind both GALE Arabic treebanks is producing more data to serve GALE targets such as improved machine translation using syntactic models. As a result of these efforts, the GALE community now has available to it a variety of morphologically and syntactically annotated Arabic data to use.

This section is divided into two parts describing these two GALE efforts.

1.3.4.2. Upgrading and Enhancing the Penn Arabic Treebank

Authors: Mohamed Maamouri, Ann Bies, Seth Kulick

1.3.4.2.1. Introduction

The LDC Arabic Treebank team has significantly revised and enhanced its annotation guidelines and annotation procedures during GALE. The revision process was initiated by the GALE sponsor and research teams based on lower than expected initial parsing scores and a high level of inconsistencies in the annotation of the Arabic Treebank (ATB)

data. Those inconsistencies were the initial targets for improvement in both guidelines and annotator training.

In order to reduce the cognitive load on annotators, both morphological and syntactic annotation guidelines were revised in several respects to be more closely aligned with traditional grammar concepts already familiar to our annotators. We also determined that many of the problems derived from an improper partitioning of the work between different levels, both conceptually and in the actual annotation procedure. Conceptually, subordinating syntactic to semantic needs in certain constructions led to inconsistencies in annotation, as different annotators gave higher priority to one or the other. In order to address such concerns, the decision was made to subordinate semantic needs to syntactic needs in constructions such as إضافة/idafa (noun-noun construct state structures) with quantifiers. Overhauling some key aspects of the morphological/part-of-speech guidelines, such as making new, more fine grained tags for comparatives, quantifiers and numerals, provided more information to the Treebank annotators and to the parser (Maamouri *et al.* 2008).

The annotators then underwent a period of intensive training focused on the revised annotation guidelines and difficult linguistic structures, as well as on consistency in general. This training combined with the revised guidelines has resulted in a markedly higher rate of inter-annotator agreement and a more consistently annotated corpus.

We have now completed automatic and partial manual revisions to all of ATB1⁵, ATB2⁶ and ATB3⁷, bringing them into line as far as possible⁸ with the new annotation guidelines and greatly improving the annotation consistency and parsing results. The success of this revision process and the improved outcome has been the result of a significant collaboration between data providers and end users.

1.3.4.2.2. Motivation for Enhanced Annotation

The revision process was initiated based on lower than expected initial parsing scores and on an examination of inconsistencies in the annotation. Parser scores for a statistical parser trained on ATB data were roughly 9 points in absolute f-measure below that of the English Penn Treebank (WSJ) (see section 1.3.4.2.7 for more detail on these scores). Inconsistencies within the Treebank annotation regarding the relationship between Morphological/Part-of-Speech (POS) tags and the syntactic annotation as well as inconsistencies in the annotation of certain syntactic constructions were shown to contribute to the parser performance. Those inconsistencies were, therefore, the initial targets for improvement both in the guidelines and in annotator training.

Many of the inconsistencies derived from an improper partitioning of the work between different levels, both conceptually and in the actual annotation procedure. Conceptually, subordinating syntactic to semantic needs in certain constructions led to inconsistencies in annotation, as different annotators gave higher priority to one or the other. For example, a quantifier-noun sequence such as “every collection” in Arabic is traditionally

⁵ LDC2008E61 - Arabic Treebank Part 1 v 4.0

⁶ LDC2008E62 - Arabic Treebank Part 2 v 3.0

⁷ LDC2008E22 - Arabic Treebank Part 3 v 3.1

⁸ As far as possible within the current time constraints and without a full-scale entirely manual review.

expressed in terms of an إضافة/idafa construction, in which the noun is considered dependent on the quantifier, which itself is treated as a noun:

```
(NP every/all/each_one | -kul~u | - كُلُّ
  (NP collection/group | majomuWapK | مَجْمُوعَةٌ))
```

However, in earlier ATB work, this structure was treated as flat

```
(NP every/all/each_one | -kul~u | - كُلُّ
  collection/group | majomuWapK | مَجْمُوعَةٌ)
```

in order to make what is often thought of as the “semantic head” (here, “collection”) more easily accessible to users. However, annotators applied both interpretations, and such structures were inconsistently annotated in the Treebank. The resolution of this type of inconsistency among others led to substantial revision of the annotation guidelines.

1.3.4.2.3. Improvements to Annotation Guidelines and Procedures

More complete and detailed annotation guidelines overall were developed, and a period of intensive annotator training focusing on the new guidelines and on specific inconsistently annotated constructions followed.

Both POS and Treebank guidelines were revised in several respects, balancing the goals of (1) representing more finely-grained distinctions, and (2) aligning more closely with traditional grammar concepts already familiar to annotators.

1.3.4.2.4. Morphological/Part-of-Speech Level

We overhauled some key aspects of the POS guidelines, such as making new tags for comparatives and quantifiers, since tagging these simply as NOUN was not informative enough for either the Treebank annotators or the parser. The POS tags for nouns and adjectives in particular were revised to be more fine-grained. In addition to NOUN_PROP (proper name), the core POS tag of NOUN is now further distinguished as

- NOUN (common noun)
- NOUN_NUM (number)
- NOUN_QUANT (quantifier)

The core POS tag of ADJ is also further distinguished as

- ADJ (common adjective)
- ADJ_NUM (ordinal number)
- ADJ_COMP (comparative adjective)

The above greater distinctions among nouns and adjectives also follow traditional Arabic grammar categories.⁹

Additional POS changes were also made to more closely follow traditional Arabic grammar categories, for example, the number of prepositions was drastically reduced (most of those lexical items are now categorized as NOUN), and many particles are now given several POS alternatives, again closely aligned with traditional categories. For example, the word *fa* had one POS value only in previous Treebank annotation: CONJ. The word *fa* now has four different POS tags available, corresponding to potentially different syntactic contexts and following its four traditional categories: CONJ (for *fa* *Al-EaTf*/ فاء العطف , the *fa* of coordination), CONNEX_PART (for *fa* *Al-rabT*/ فاء الربط , the *fa* of connection), RC_PART (for *fa* *Al-jazA*'/ فاء الجزاء , the *fa* of reward, response conditional), and SUB_CONJ (for *fa* *Al-sababiy~ap*/ فاء السببية , the *fa* of causality).

A new POS category of pseudo-verbs has been added to account for the verbal behavior of certain Arabic particles. These are “the sisters of *inna*” (with the exception of *anna*,” the complementizer “that”), a category regarded by Arabic grammarians as having verbal properties, such as subcategorizing for a subject and a predicate or clausal complement. Since these words display verbal behavior although they are not technically verbs, they will now be given the POS tag “PSEUDO_VERB” and head a VP in the tree.

1.3.4.2.5. Syntactic/Treebank Level

In order to address concerns such as the inconsistent annotation of quantifiers, the decision was made to subordinate semantic needs to syntactic needs in certain constructions (for example, *idafa* with quantifiers).

As the *idafa* structure is a particularly frequent noun phrase structure, this decision affects the annotation of a significant portion of the corpus. In *idafa* structures syntactically headed by common nouns, the semantic and syntactic head of the noun phrase will be the same noun (as in the “grammar book” example below, where “book” is both the semantic and the syntactic head of the noun phrase).

```
(NP (NOUN+CASE_DEF_NOM
      كتاب kitaAbu book)
  (NP (NOUN+CASE_INDEF_GEN
      نحو naHowK grammar)))
  كتاب نحو
  (a) grammar book
```

vs.

```
(NP (NOUN_QUANT+CASE_DEF_NOM
      every -kul~u - كُلُّ )
  (NP (NOUN+NSUFF_FEM_SG+CASE_INDEF_GEN
      collection majomuwEapK مَجْمُوعَةٌ)))
  كُلُّ مَجْمُوعَةٍ
  every collection
```

⁹ For now we have chosen to retain the noun and adjective labels for active participles, passive participles and gerunds/masdars, rather than switching to an entirely traditional representation of these categories.

However, in *idafa* structures that are syntactically headed by quantifiers (as in the “every collection” example above), the semantic head of the noun phrase is not the quantifier at all, but its dependent noun. The interaction of this *idafa* structure with the new, more fine-grained POS tags allows the difference in semantic and syntactic heads to be captured. The syntactic/Treebank annotation is based on the syntactic head (the quantifier, “every”). However, the semantic head (the dependent noun, “collection”) is still easily accessible to end-users based on the POS tag NOUN_QUANT on the quantifier. This interaction of the changes in POS and syntactic annotation guidelines also results in an overall conceptual and practical improvement, since (1) a simple algorithm can recover the necessary semantic information, and (2) inter-annotator agreement is higher.

As with the revision of POS guidelines, the revision of the syntactic annotation guidelines also served to more closely align the Treebank annotation with traditional Arabic grammar categories for several constructions. These include the treatment of comparatives, numbers and numerical expressions and the treatment of several particular pronominal constructions such as separating pronouns/Damiyr Al-faSI/ضمير الفصل and anticipatory pronouns/Damiyr Al\$>n/ضمير الشأن .

Further revisions include a more careful and complete classification of verbs and their argument structure and a more comprehensive approach to the contexts in which gerunds or participles have a verbal reading. For example, a *masdar*/gerund, active participle or passive participle followed by a PP complement to the regular verb form (PP-CLR) is now shown with a verbal reading.

For a more complete description of the new annotation policies, see the *Arabic Treebank Morphological and Syntactic Annotation Guidelines* (2008).¹⁰

1.3.4.2.6. Corrections of Previous Annotation Level

The initial POS annotation is still selected from the morphologically analyzed alternatives provided by the Buckwalter morphological analyzer (BAMA 2004).¹¹ However, crucial to reducing the number of mismatches between POS tags and syntactic structures is the ability of Treebank annotators to correct certain specified POS tags from the earlier annotation level. The annotation tool has been revised so that Treebank annotators now have the ability to correct specific POS tags such as CONJ → ADV or PREP → NOUN. In addition, a careful set of quality control searches based on head rules leads to further manual correction of POS tags in conflict with the tree structure.

1.3.4.2.7. Improvements in Inter-annotator Agreement and Training

Intensive annotator training focused on agreement and consistency and led to an improvement of inter-annotator agreement scores on the tree structures from an initial f-measure of 86.98% to the much improved f-measure of 94.3%.

The ATB production workflow includes both automatic and manual error correction, along with planned on-going annotator training, and it is hoped that these measures will continue to improve the agreement further. In order to maintain a high rate of inter-

¹⁰ <http://projects ldc.upenn.edu/ArabicTreebank/>

¹¹ Future new annotation will use the newly revised LDC Standard Arabic Morphological Analyzer (SAMA) Version 3.1. LDC Catalog Number: LDC2009E73.

annotator agreement, approximately 10% of each corpus is dual blind annotated during production and put through the full workflow. The 94.3% score measured 293 sentences of dual blind annotation from the ATB3 (Annahar) corpus, annotated independently by two Treebank annotators. This score was computed using the standard evalb program (not including function tags).

The initial agreement score was considered to be too low for the purpose of training statistical parsers on the ATB data. The goal was to approach the reported score of 93.8% for the Chinese Treebank. This goal has now been not only met but surpassed, and data produced with this level of agreement is expected to support on-going work on improving parsing results.

1.3.4.2.8. Revision Process and Quality Assurance

As described above, the annotation process for creating the original corpora consisted of a level of POS/morphological annotation on the original tokens from the text file, followed by separation of various clitic morphemes to create the tokens actually used for treebanking.

The guideline revisions specified changes at various different levels, including tokenization and POS tags as well as the trees. As a result, there are cases in which different decisions are now being taken in the morphological guidelines as to when the original tokens should have been split or not. In order to modify the tokenization to match the current guidelines, it was not possible to do so only by examining individual tokens in the Treebank, since such tokens may themselves be part of a larger original token.

For example, while “limA*A” formerly existed in the ATB3-v2.0 corpus both as a single token and also split into two tokens (“li” and “mA*A”), in the revised morphological guidelines it is now treated as one token only. However, the annotation as it existed in the ATB3-v2.0 corpus for the two-token analysis had already split up the word, and the individual Treebank tokens “li” and “mA*A” were both acceptable tokens unto themselves. It was only in the context of being part of a larger original word that it could be recognized that they needed to be merged back together for this revised release.

Therefore, we created a version of the corpus which associated each original token from the source text file with the one or more Treebank tokens that together make up that original token.¹² We then modified the tokenization automatically based on that correlation. This also helped to identify possible tags in some cases for when the range of POS tags was restricted when part of a larger token (e.g., “mA” in the context of “bi+mA” has a smaller range of possible tags compared to “mA” occurring by itself).

The software written for this process was not only useful for automatic changes to adjust the tokenization, but also resulted in a characterization of all original tokens for, roughly speaking, the closed class “function words” that were the focus of the POS revisions. That is, all such tokens were automatically identified in terms of potential component morphemes and possible POS tags for each morpheme (e.g., the “bimA” case just mentioned). This helped ensure the consistency and accuracy of annotation at the token level.

¹² We found this version to be so useful internally that as of ATB5, ATB releases will also include a version of the corpus showing this mapping between the source text tokens and the Treebank tokens.

In addition to automatic revisions as above, we significantly improved the post-annotation quality control (QC) process for the ATB. The QC process consists of a series of specific searches targeting several types of potential inconsistency and annotation error, and we increased the number of error searches threefold during the ATB revision process. These error searches are run after annotation is complete, and any errors found via these searches are hand corrected.

1.3.4.2.9. Parsing Improvement and Analysis

An important goal is to evaluate the increase in parser accuracy as a result of the revisions described in this section, and to compare the current accuracy to that of parsing on a more established source, namely the Wall Street Journal portion of the English Penn Treebank (PTB). Using a previously proposed data split, we trained and tested on each of the revised ATB1, ATB2 and ATB3 individually, as well as the combined ATB123 (738, 845 tokens/words in total), for both the newly revised versions and the older pre-revision releases.¹³ In addition, we have trained and tested on an amount of PTB data comparable in size to ATB3, since ATB3 is the largest of the three revised ATB corpora, as well as for the combined ATB123.

The parser used was the Bikel adaptation of the Collins parser.¹⁴ We ran the parser in two modes. In both, the parser input contains the gold Part-of-Speech tags. The dev section results in Table 1.10 columns 2-4 are for the mode in which the parser used the given tags only for words with which it was unfamiliar from training, and otherwise was free to choose its own tags. In the second mode, shown in Table 1.10 columns 5-7, the parser was forced to use the given tag for each word.

Tags	Parser's own			Supplied		
	Old	New	PTB	Old	New	PTB
ATB3	77.5	81.0	87.6	78.5	83.2	87.2
ATB123	78.8	82.7	88.6	79.1	84.1	88.8

Table 1.10: Parser Results, with parser choosing its own tags and with parser forced to use given tags.

As can be seen in Table 1.10, there is improvement in the score for the parser with the revised data, roughly halfway bridging the gap to the PTB score.¹⁵ It is perhaps of note as well that there is a greater distinction in the two ways of running the parser for ATB as compared to PTB. This is perhaps indicative of greater tree/tag consistency in the ATB, or perhaps of a greater share of the burden put on the pos tags. This is a matter for further study, but in both parser modes there is noteworthy improvement for the new compared to the old scores.

In order to better understand the source of the parser improvement, we performed a dependency analysis, as was also done in Kulick, Gabbard and Marcus (2006). Each parser output tree and corresponding gold tree is broken down into a collection of relations, which is a one-level slice of the context-free tree. We have selected some of the

¹³ <http://nlp.stanford.edu/software/parser-arabic-data-splits.shtml>

¹⁴ <http://www.cis.upenn.edu/~dbikel/software.html#stat-parser>

¹⁵ Note, however, that due to differences in data and style of annotation, a comparison of scores across Treebanks is at best a rough indication of performance differences.

most frequent relations for ATB123 and categorized them into two groups, shown in Table 1.11. In this table, the columns are (1) the relation, (2) the frequency of that relation in the new ATB123, and (3-5) the scores for ATB123-old, ATB123-new, and PTB.

The top portion of Table 1.11 shows relations that make up what might be called the core syntactic structures. For example, the relation NP \rightarrow NOUN NP is the idafa construction.¹⁶ PP \rightarrow PREP NP is the relation of the object of a preposition to the PREP, and so on. A “base NP” (NPB) is an NP without another NP inside it. This table demonstrates the improvement in the parser recovery of these core relations, which seems strongly indicative of increased Treebank internal consistency. One place where there is certainly room for improvement is with the S \rightarrow NP VP relation, in which it seems likely that the parser is getting confused over the optionality of the subject placement in Arabic.

Relation	% of all relations in ATB123-new	ATB123-old f-measure	ATB123-new f-measure	PTB f-measure
NP \rightarrow NOUN NP	16.75	90.4	97.4	n/a
PP \rightarrow PREP NP	13.40	96.5	99.2	95.2
Base NP	12.71	84.1	90.2	95.0
VP \rightarrow verb NP	11.59	92.1	94.1	93.3
SBAR \rightarrow compl S	2.59	91.1	92.9	92.0
S \rightarrow NP VP	2.03	87.4	91.3	96.3
VP \rightarrow VERB PP	6.44	82.6	83.4	83.5
NP \rightarrow NPB PP	3.49	73.3	75.7	86.2
NP \rightarrow NP PP	1.77	33.5	45.0	n/a

Table 1.11: Parser accuracy on core syntactic structure relations (top) and PP attachment relations (bottom).

The bottom portion of Table 1.11 shows the three relations having to do with PP attachment. Here, while the score for attachment of a PP modifier to a VP is nearly identical to that of the PTB, the score is significantly lower for PP attachment to an NPB (the very low scoring relation for PP attachment to an NP that is not an NPB does not even exist in the PTB). The lower score for the NP \rightarrow NPB PP relation is no doubt because of the impact of the idafa structure upon the PP attachment problem, as has been discussed in the literature (see e.g., Kulick *et al.* 2006 and Gabbard and Kulick 2008).

1.3.4.2.10. Conclusions

We have discussed some of the issues that arise when the Arabic Treebank syntactic annotation is manually enhanced as the first step, ahead of the morphological/Part-of-Speech annotation. We outlined an automatic procedure that more closely aligns the POS tags and the Treebank annotation, leading to increased parsing results and additionally providing the annotation pipeline with improved error checking and quality control. The importance of the interaction between POS tags and the tree structure is shown by the increase in parsing results obtained by forcing the parser to use the given tags resulting from this procedure, also indicating the important role that a POS tagger would play in a

¹⁶ Note that this right branching structure is also used for tamyiz in some constructions. See the ATB syntactic guidelines for details on those constructions.

full Arabic NLP pipeline. In future work, we intend to investigate whether certain of the tags may be more crucial for the parser to get right. It seems reasonable that many of the “function word” particles are particularly crucial.

Revising the annotation of the existing Arabic Treebank corpora to reflect the newly updated guidelines has provided a significantly improved resource to the community. Additional annotation of new data in the improved guidelines style will follow.

The improved ATB guidelines, improving inter-annotator agreement scores, and an expected continuing improvement in parsing scores are the result of a fruitful collaboration between data producers, sponsors and end users, along with the support and time to effect the change. It is hoped that such collaboration will continue to benefit both annotation production and NLP applications in the future.

1.3.4.3. The Columbia Arabic Treebank

Authors: Nizar Habash, Reem Faraj and Ryan Roth

1.3.4.3.1. Introduction

Under time restrictions, the creation of a treebank faces a tradeoff between linguistic depth and treebank size. This is especially the case for morpho-syntactically complex languages such as Arabic or Czech. Linguistic depth provides the advantage of providing many linguistic features that may be useful for a variety of applications. This comes at the cost of slower annotation as a result of longer guidelines and more intense necessary annotator training. As a result, the deeper the annotation, the slower the annotation process and the smaller the size of the treebank. And consequently, the less data there is to train tools that can benefit from more data.

Two basic ideas inspire the Columbia Arabic Treebank (CATiB) approach. First, CATiB avoids annotation of redundant linguistic information. For example, nominal case marks and state (definite, indefinite, construct) in Arabic are determined automatically from syntax and morphological analysis of the words and needn’t be specified by human annotators. Of course, some information is not easily recoverable in CATiB, such as phrasal co-indexation and full lemma disambiguation. Second, CATiB uses a linguistic representation and terminology inspired by Arabic’s long tradition of syntactic studies. This makes it easier to train annotators. CATiB uses an intuitive dependency representation and relational labels inspired by Arabic grammar such as *idafa* (a syntactic construction used for indicating a possession relationship between two nouns) in addition to the universally recognizable predicate-argument labels of *subject* and *object*.

1.3.4.3.2. CATiB Linguistic Annotation Profile

CATiB uses the same tokenization scheme of the PATB and PADT. However, unlike these resources, the CATiB POS tag set is much smaller. Whereas PATB uses over 400 (tokenized) tags specifying every aspect of Arabic word morphology such as definiteness, gender, number, person, mood, voice and case; CATiB uses six POS tags: NOM (nominals such as nouns, pronouns, adjectives and adverbs), PROP (proper noun), VRB (verb), VRB-PASS (passive verb), PRT (particles such as prepositions or conjunctions) and PNX

(punctuation). CATiB uses a dependency representation that models predicate-argument structure (subject, object, etc.) and Arabic nominal structure (idafa [possessive construction], tamyiz [specification construction], modification). Here are all the relations: SBJ (subject of verb or topic of simple nominal sentence), OBJ (object of verb, preposition, or deverbal noun), TPC (topic in complex nominal sentences containing an explicit pronominal referent), MOD (general modifier of verbs or nominals), IDF (idafa in nominal constructions) and TMZ (tamyiz in nominal constructions). No empty categories are made explicit and no co-indexation of phrases. A detailed discussion of the CATiB guidelines is presented by Habash *et al.* (2009). Although the CATiB tag set is small, it is appropriate for human annotation purposes. We are able to reproduce a parsing-tailored tag set [size 36] (Kulick *et al.* 2006) automatically at 98.5% accuracy using features from the annotated trees. Details of this result will be presented in a future publication.

1.3.4.3.3. CATiB Annotation Process

Although CATiB is independent of previous annotation projects, we made sure to build on top of existing resources and lessons learned. In our CATiB pipeline, we tokenize using the MADA+TOKAN toolkit, trained on PATB (Habash and Rambow 2005). We initially parse using the MALT parser trained also on a converted version of the PATB (Nivre *et al.* 2007). The generated trees are then manually checked and modified using the Tred toolkit produced by the team that developed the PADT (Pajas 2002).

Our annotators are all educated native Arabic speakers – although not necessarily in Arabic linguistics or linguistics in general. The training process took place over two months (150 hours on average). This is much faster than the reported six months training for PATB. The speed of annotation of POS and syntax in production mode is over 540 tokens/hour. The speed of verifying the tokenization is over 5K words/hour. In PATB annotation, the reported speed of syntactic annotation alone (no POS) is similar to CATiB's. When combining this speed with the speed of the PATB POS annotation step, we get an effective 250-300 tokens/hour – almost half the CATiB speed.¹⁷

We were able to create the CATiB manual from scratch; interview, hire and train annotators; and produce over 228K annotated words all in the span of seven months. In addition to annotation of new data, we automatically converted PATB trees to our CATiB representation – this task is possible primarily because we use less linguistic information. Collectively, the CATiB version 1.0 release contains over 1M tokens (841K words, 31,319 trees), including annotated and converted data.

Our basic inter-annotator agreement scores are around 91.6% on unlabeled parent attachment (for sentences ≤ 40 tokens) and reach up to 94.7%. A serial annotation experiment, where the annotators corrected already-annotated output, gives results reaching up to 97.8% at 30% overall speed reduction. These results are generally comparable to previous efforts that take longer time to train and annotate. More details about CATiB results are presented by Habash and Roth (2009).

¹⁷ Personal Communication with Mohamed Maamouri.

1.3.5. OntoNotes: A Large Training Corpus for Enhanced Processing

Authors: Ralph Weischedel, Eduard Hovy, Mitchell Marcus, Martha Palmer, Robert Belvin, Sameer Pradhan, Lance Ramshaw, Nianwen Xue

1.3.5.1. Introduction

This section describes a large multilingual richly annotated corpus that is being made available to the community. We emphasize quality and consistency, targeting interannotator agreement rates at 90%. The data covers multiple genres in English, Chinese, and Arabic, including a significant amount of parallel data. The annotation, intended to capture a skeletal representation of literal meaning, includes parse trees, predicate argument structures, word senses localized in an ontology, coreference, and name types. The resource is delivered as an integrated database, supporting combined queries that access multiple annotation layers. Annual incremental releases are distributed via the LDC.

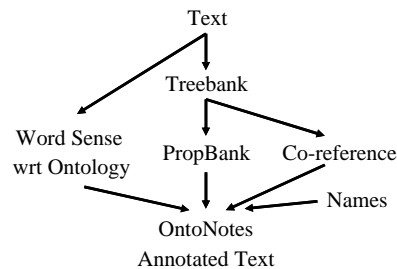


Figure 1.7: Levels in OntoNotes.

1.3.5.2. Motivation, Goals, and Rationale

As shown in Figure 1.7, to the baseline structure of parse trees and propositions, OntoNotes adds:

- Referring expressions and the textual phrases they refer to
- Terms disambiguated by word sense and localized in an ontology
- Named Entities

Based on our interpretation of the criteria for successfully applying learning algorithms, our guiding principle has been to find a “sweet spot” in the space of

- Inter-tagger agreement, so that human agreement as a ceiling on algorithm performance is as high as possible.
- Productivity, so that the amount of training data is maximized, given a budget,
- Depth of representation, so that the added semantic features are as deep as possible.

The methodology described here was tested prior to entering production mode, where pilot rounds of annotation were conducted to find the sweet spot above. In particular, only those classes of co-reference satisfying the methodology above during the pilot study are annotated. The methodology has been applied for word sense, with the sense inventory for each word being selected according to the criteria above.

Another dimension of the OntoNotes product is the integration of all of the annotations in a database (Pradhan *et al.* 2007a), which has at least two benefits:

- Consistency checks on entering each annotation element flag many inconsistencies across annotations for manual correction.
- The data may be searched for phenomena of interest.

This section illustrates annotation primarily of English, though OntoNotes covers Arabic and Chinese as well. In the next sections, we describe each of the component annotations: treebanking, proposition banking, word sense, ontology creation, coreference, and names. The section concludes with a summary of related work.

1.3.5.3. Treebanking

The Treebank style in OntoNotes for English is a modification of the Treebank II style for the Penn Treebank (Marcus *et al.* 1993; Marcus *et al.* 1994). For Chinese, the style follows the Chinese Treebank (Xue *et al.*, 2005). (Arabic Treebank annotation is being performed at the Linguistic Data Consortium.) These are annotated with information to make predicate-argument structure easy to decode, including function tags and markers of “empty” categories that represent displaced constituents.

To facilitate merging of the syntactically annotated material with the PropBanked material, both the Treebank style and PropBank style were modified to correct for some small mismatches between the annotations (Babko-Malaya *et al.* 2006). The major changes to the Treebank stylebook involved modifying the list of verbs considered to take so called “small clauses” to conform to the argument structures assigned by PropBank, and changing the structures of resultatives to match the PropBank analysis.

The internal consistency of the newly syntactically annotated material for English has been tested, and is quite good. The principal annotator for English reannotated sampled material a year after the original annotation. The F-measure of the newly annotated material against the initial annotation by the EVALB measure was 98.5.

A major editing pass of the OntoNotes Treebank materials is now underway to achieve full consistency with all materials treebanked under the GALE project. The first modification retrofits much of the OntoNotes treebanked materials to conform with the current LDC syntactic style for NPs, crucially adding branching structure whenever the default right branching structure of pre-head modifiers is violated. The second modification changes the tokenization so as to split on most token-internal hyphen. This eliminates some anomalies that resulted from the earlier tokenization, which was based entirely on white space. In “the New York-based company”, for example, the old tokenization, with “York” attached to “-based” made it impossible to annotate “New York”.

1.3.5.4. PropBanking

PropBanking focuses on annotating the argument structure of verbs, and provides a corpus annotated with semantic roles, including participants traditionally viewed as arguments and adjuncts. The style for English is that of the 1M word Penn Treebank II Wall Street Journal corpus (Palmer *et al.* 2005). In addition to annotating verbs we are also applying Nombank style annotation to just those nouns with predicate-argument structures that can participate in event coreferences, such as nominalizations and eventive nouns. Links from the argument labels in the Frames Files to FrameNet frame elements and VerbNet thematic roles have been added. This style of annotation has also been successfully applied to other genres and languages. For Chinese, the style is that of (Xue & Palmer 2009) The same style has also been applied to Arabic (Diab *et al.* 2007).

1.3.5.5. Word Sense

One of the daunting challenges was attaining 90% annotator agreement for word sense, since, for example, WordNet inter-annotator agreement averages in the low 70s. Building on results in grouping fine-grained WordNet senses into more coarse-grained senses that led to improved inter-annotator agreement (ITA) and system performance (Palmer *et al.* 2007), we have developed a process for rapid sense inventory creation and annotation that includes critical links between the grouped word senses and the Omega ontology (Philpot *et al.* 2005).

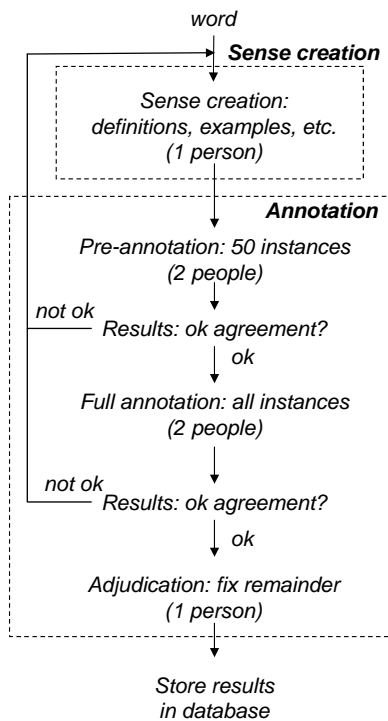


Figure 1.8: Annotation Procedure

Figure 1.8 shows the empirical process for proposing meaningful sense distinctions and determining if they could be annotated at 90% accuracy. A 50-sentence sample of instances is annotated and immediately checked for inter-annotator agreement for all verbs and any noun with frequency over 100. ITA scores below 90% lead to a revision and clarification of the groupings by the linguist. It is only after the groupings have passed the ITA hurdle that each individual group is linked to a conceptual node in the ontology. In addition to higher accuracy, we find at least a three-fold increase in annotator productivity.

The same methodology has been applied to English, Arabic, and Chinese; the only difference is the starting point for suggesting sense inventories. For English, WordNet has been our starting point of choice. For Chinese, diverse sources are reviewed before hypothesizing an inventory, including entries in web-accessible dictionaries, print dictionaries, samples from the corpora to be annotated and general web searches. For Chinese verbs, the starting point has been the course-grained senses in the frame files created for the PropBank annotation, although print and electronic dictionaries are also consulted. Similar research including access to Arabic WordNet is carried out for each Arabic word prior to authoring its sense inventory file.

1.3.5.6. Verbs

G1: operating or traveling via a vehicle <i>NP (Agent) drive NP, NP drive PP</i>	WN1: "Can you drive a truck?", WN2: "drive to school," WN3: "drive her to school," WN12: "this truck drives well," WN13: "he drives a taxi," WN14: "The car drove around the corner," WN:16: "drive the turnpike to work,"
G2: force to a position or stance <i>NP drive NP/PP/infinitival</i>	WN4: "He drives me mad.," WN5: "She is driven by her passion," WN6: "drive back the invaders," WN7: "She finally drove him to change jobs," WN15: "drive the herd," WN22: "drive the game."
G3: to exert energy on behalf of something <i>NP drive NP/infinitival</i>	WN11: "What are you driving at?," WN10: "He is driving away at his thesis."
G4: cause object to move rapidly by striking it <i>NP drive NP</i>	WN9: "drive the ball into the outfield ," WN17 "drive a golf ball," WN18 "drive a ball"

Table 1.12: Four Groups for "drive", Compared to the WordNet Senses

The word sense inventories for English verbs come initially from grouping related WordNet senses (Palmer, et. al. 2007). Subcategorization frames and semantic classes of arguments play major roles in determining the groupings, as illustrated by the grouping for the 22 WN 2.1 senses for *drive* in Table 1.12. The groupings are also linked to the ontology (see Section 1.3.5.5). In addition to improved annotator productivity and accuracy, we have found a corresponding improvement in word sense disambiguation performance. Training on this new data, Dligach and Palmer (2008) report 83% accuracy for verbs using a Support Vector Machine and rich linguistic features, which is almost 20% higher than state-of-the art performance on ungrouped, fine-grained senses (Chen and

Palmer, 2005). The sense inventories for Chinese and Arabic verbs are created by starting with the PropBank frame files and subdividing the verb entries into more fine-grained senses where deemed appropriate.

1.3.5.7. Nouns

Noun annotation follows a procedure similar to that for verbs. The noun senses are created starting with WordNet and other dictionaries. We aim to double-annotate, at the target agreement level, the 1100 most frequent polysemous English nouns in the entire corpus before the end of 2009, while maximizing overlap with the sentences containing annotated verbs. We have lower targets for the other two languages, which were started later.

Certain nouns carry predicate structure. To ensure conformity with verbs, the structure of nominalizations (*destruction*) and eventive nouns (*party*) is created and assigned by the verb specialists at Colorado.

In order to speed up annotation, we investigated a form of active learning, in which nouns with high agreement in a subset of the whole corpus were used as training data by an automated annotation learner. Unfortunately, different sense distributions across corpora meant that we could not always use results from one year to automatically annotate another year's data. We investigated various strategies to bootstrap the learning, by mixing into the training data small amounts of annotated data from the new corpus. The results show that even 50 instances from the new distribution permit learning that is accurate enough for about 50% of the high-frequency nouns (Zhu and Hovy 2007; Zhu *et al.* 2008).

The OntoNotes (release 1.0) verb and noun word sense data was used in the Semeval-1 (Pradhan *et al.* 2007b). Overall accuracy over 100 lemmas (65 verbs and 35 nouns) from WSJ corpus, for the best performing system was 86% — with average over verbs being 78% and over nouns being 89%.

1.3.5.8. Coverage Issues

There are far too many polysemous lexical items for any project to provide exhaustive coverage. Therefore the prioritization of items for annotation is of pressing concern.

Clearly high frequency items provide the most leverage, but they often have a predominant sense (as much as 90% of the data) which can overwhelm annotators with hundreds or even thousands of repetitive examples that will provide little if no system performance improvement. For example, 183 of the 186 instances of the word “bank” in the Ontonotes portion of the WSJ corpus are cases of the first of the 10 senses (a financial institution). In all corpora combined, 607 of the 640 instances of “investment” are the third sense (the activity of investing money for profit). With this type of data, double blind manual annotation and adjudication is not really necessary.

Of course, prior to the manual annotation the entropy of a word's sense distribution is unknown. When we are partially through annotation of a given word and it is clear that the majority of its instances fall into one sense, we can then dispense with full double-annotation of all senses, and allow one of the “annotators” to be a trained classifier. This method is described in more detail by Zhu and Hovy (2007). This allows for

somewhat quicker annotation progress to be carried out for the most common senses, so more time can be devoted to human annotation of rarer words and senses.

The desired aim is a balance between sufficient coverage of high frequency items and maximal coverage of low frequency ones. For these rare words and senses the greatest challenge is finding enough instances to provide adequate training material. We are also exploring techniques such as language modeling for preselecting instances of rare senses from a new corpus (Dligach and Palmer 2009). In addition we have implemented a data selection plan which supplements our “whole document” based annotation approach with lexical samples for specific lexical items which require greater coverage.

1.3.5.9. Ontology

Standard dictionaries simply list the senses for each word. To support synonym access, inheritance of features and other properties such as predicate frames, links to instances, and so on, we group together the senses that share the same meaning, and then arrange them into a shallow taxonomy that we call the Omega Ontology (Philpot *et al.* 2005) following the process of Figure 1.9.

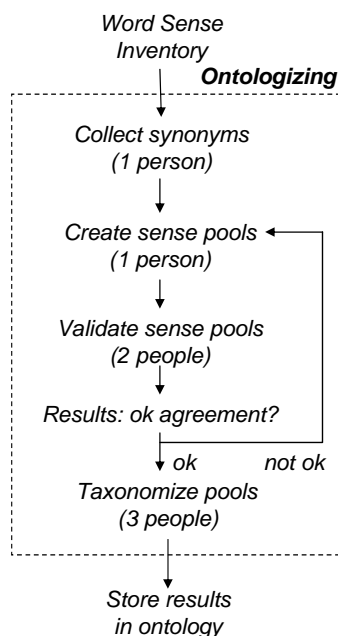


Figure 1.9: Process for adding to the Ontology.

A manual procedure forms *sense pools* by selecting and grouping together individual noun and verb senses that convey the same meaning. (Sense pools correspond to WordNet’s synsets, but are generally less fine-grained.) Each sense pool contains one or more definitions, examples, features, and pointers to the individual senses that comprise it, from which one can access their respective annotated sentences. It is thus possible to

assemble for each meaning a set of sentences that contain different target words, each expressing that meaning, in order to train more-powerful sense disambiguation engines.

All sense pools are attached into Omega's Upper Model (Hovy *et al.* 2009), a network of some 120 nodes that represent very abstract conceptualizations. Reference to VerbNet semantic classes has been helpful in creating nodes for the verb upper level ontology (Palmer *et al.* 2009). To date, about 5000 noun-derived and 3500 verb-derived pools (from English sources) have been created and attached, by multiple annotators who compare their decisions to ensure quality, using a specialized interface. Work is underway to create sense structures also for the other two languages and either merge them into English-derived pools or attach them to the Upper Model separately. In addition, we have started creating sense structures for the 3000-odd monosemous English nouns occurring in the corpus, and merging or inserting them into Omega.

1.3.5.10. Coreference

The coreference annotation in OntoNotes connects coreferring instances of specific referring expressions, primarily NPs that introduce or access a discourse entity. For example, "Elco Industries, Inc.", "the Rockford, Ill. Maker of fasteners", and "it" could all corefer. (Non-specific references like "officials" in "Later, officials reported..." are not included, since coreference for them is frequently unclear.) In addition, proper noun pre-modifiers and verb phrases can be marked when coreferent with an NP, such as linking, "when the company withdrew from the bidding" to "the withdrawal of New England Electric".

Unlike the coreference task as defined in the ACE program, attributives are not generally marked. For example, the "veterinarian" NP would not be marked in "Baxter Black is a large animal veterinarian".

However, the sense of "be" is marked so that attributive information is annotated. Adjectival modifiers like "American" in "the American embassy" are also not subject to coreference.

Appositives are annotated as a special kind of coreference, so that later processing will be able to supply and interpret the implicit copula link.

All of the coreference annotation is being doubly annotated and adjudicated. Over the first two years, the overall average agreement between individual annotators and the adjudicated result for non-appositive coreference using the MUC coreference scorer was 86%.

Pradhan *et al.* (2007) report baseline performance on the OntoNotes coreference data using a standard feature set. Coreference decoding contrasts with decoding other layers in that system performance on coreference still lags very much behind the ITA, in spite of the latter being very high. This is most likely due to the fact that richer semantic and word-knowledge components, in addition to annotation granularity and consistency, are important in identifying co-referring entities. Better learning strategies combined with the accompanying layers in OntoNotes would likely help bridge this gap in the future.

1.3.5.11. Names

Names are also annotated using an 18-type superset of the ACE name guidelines. This supplemental annotation is done in a single pass.

1.3.5.12. Database

Since we are delivering multiple levels of annotation (syntax, propositions, coreference, word sense, ontology, and names), several questions arose:

1. How could we ensure that all the components are consistent with each other, avoiding engineering/formatting inconsistencies?
2. Should the annotations be delivered as independent pieces provided in an integrated representation?
3. What representation would best facilitate use of this information as training data for systems that will be incorporated into applications? Can this representation also support leveraging these additional knowledge sources during the training process?

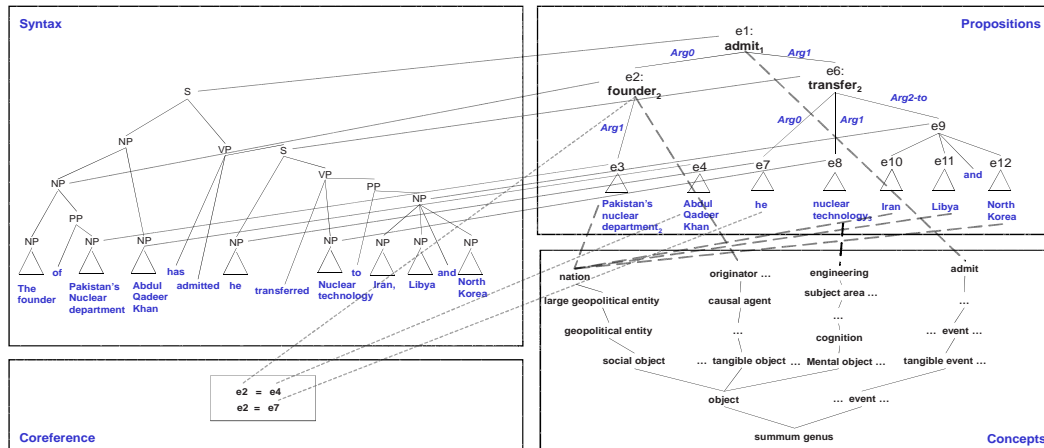


Figure 1.10: Simplified diagram of the interconnections between annotation layers.

We have created a corpus with diverse levels of semantic information integrated in one database (Pradhan *et al.* 2007a). Figure 1.10 illustrates some of the interconnections captured. The database contains these multiple annotation levels on texts in each of the three target languages.

Each document is stored as a sequence of sentence strings. The token table identifies the word substrings by their offsets, and those tokens in turn are the leaves of a parse tree showing the syntactic structure of the sentence. Supplementary link tables implement many-to-many relations whereby PropBank, word sense, and coreference annotations can be associated with parse tree nodes. Link tables are also used to connect the OntoNotes word senses with the associated PropBank frames. For parallel text that is available in multiple languages, the database also supplies alignment information that maps from each sentence to the associated sentence(s) in the other language.

This process of integrating these disparate types of annotation into a single database identified several types of inconsistencies between the layers. Some of these were theoretical differences. For example, the original freestanding PropBank annotation used multiple pointers for a single argument whose immediate value was a trace node that pointed to another, more distant tree node. In the integrated annotation, each PropBank argument is a single syntactic constituent, since the Treebank already provides the trace information.

There were also cases where the syntactic analysis implied by the original PropBank annotation clashed with that in the Treebank. For example, for the phrase “keep their markets active”, the original PropBank analysis had “their markets” and “active” both as arguments of “keep”, while the Treebank treated “active” as the predicate of a small clause. The two groups each made some changes to resolve these inconsistencies (Babko-Malaya *et al.* 2006).

Other clashes that showed up in the database merge were just incidental, for example, tree structure changes that had been made by one set of annotators but not communicated to the other groups. Resolving all of these inconsistencies helped to ensure a clean, consistent final product, where the relationships between all the layers and within the layers themselves can be efficiently captured in the database schema.

We have also provided an object layer on top of the database layer, written in Python. This layer implements a Python object type for almost every database table. These objects directly represent data values like inverse pointers that are not stored explicitly in the database, and they thus allow for more efficient and flexible data manipulation than when working directly at the level of the underlying database tables, particularly when working across multiple layers. It can also output representations of each individual layer by itself, or a human-readable representation that combines the information in all the layers.

This object layer facilitates defining custom views of the data as well as extracting cross-layer features for use in analysis or in predictive models, neither of which was easily possible before. For example, one could use this API to find out the distribution of named entity types that occur as ARG0 arguments of the predicate “say”, combining information from the Treebank, PropBank, and names layers.

1.3.5.13. Related Work

PropBank I (Palmer *et al.* 2005), developed at UPenn, captures predicate argument structure for verbs; NomBank provides predicate argument structure for nominalizations and other noun predicates (Meyers *et al.* 2004). PropBank II annotation (eventuality ID’s, coarse-grained sense tags, nominal coreference and selected discourse connectives) has been applied to a small (100K) parallel Chinese/English corpus (Babko-Malaya *et al.* 2004). The OntoNotes representation extends these annotations, and allows eventual inclusion of additional shallow semantic representations for other phenomena, including temporal and spatial relations, numerical expressions, deixis, etc.

One of the principal aims of OntoNotes is to enable automated semantic analysis. One state-of-the-art algorithm for semantic role labeling for PropBank style annotation (Pradhan *et al.* 2005) achieves an F-score of 81.0 using an SVM model. OntoNotes will provide a large amount of new training data for similar efforts.

Other related work falls into two classes: the development of resources for specific phenomena or the annotation of corpora. An example of the former is Berkeley's FrameNet project (Baker *et al.* 1998), which produces rich semantic frames, annotating a set of examples for each predicator (including verbs, nouns and adjectives), and describing the network of relations among the semantic frames. An example of corpora annotation is the Salsa project (Burchardt *et al.* 2006), which produced a German lexicon based on the FrameNet semantic frames and annotated a large German newswire corpus. A second example, the Prague Dependency Treebank (Hajic *et al.* 2001), has annotated a large Czech corpus with several levels of (tectogrammatical) representation, including parts of speech, syntax, and topic/focus information structure. The Tsinghua Chinese Treebank TCT (Zhou 2003) contains some 2 million Chinese characters, of which half has been treebanked, and manually annotated for syntactic and certain semantic relations, such as causality and conditionals. It covers various genres

Finally, the IL-Annotation project (Reeder *et al.* 2004) focused on the representations required to support a series of increasingly semantic phenomena across seven languages (Arabic, Hindi, English, Spanish, Korean, Japanese and French). In intent and in many details, OntoNotes is compatible with all these efforts, which may one day all participate in a larger multilingual corpus integration effort.

1.3.5.14. Summary

	English	Chinese	Arabic
NW	550 K	250 K	300 K
BN	200 K	300 K	200 K
BC	200 K	150 K	–
Web	300 K	150 K	–

Table 1.13: Planned corpus (token counts).

The plan for the full OntoNotes corpus is shown in Table 1.13, covering three languages and four genres (NewsWire, Broadcast News, Broadcast Conversation, and Web text), and including significant amounts of parallel bilingual data. OntoNotes Version 2.0, released by the LDC in early 2008, covered NW and BN in English and Chinese and NW in Arabic. Version 3.0, to be released in June 2009, will add coverage of BC data in English and Chinese, with additional Arabic NW. It is our hope that this annotation will provide an enduring resource for the community.

1.4 Automatic Annotation

1.4.1. Speech Segmentation and Its Impact on Spoken Document Processing

Authors: Mari Ostendorf, Benoit Favre, Ralph Grishman, Dilek Hakkani-Tür, Mary Harper, & Dustin Hillard

1.4.1.1. Introduction

Dramatic improvements in automatic speech recognition (ASR) technology make it now possible to explore how language processing techniques designed for text can be applied to spoken language. Ever increasing collections of information are available as speech recordings, including news broadcasts, talk shows, meetings, debates, lectures, hearings, oral histories, and webcasts, among other types of human-directed (vs. computer-directed) communications. ASR can automatically transcribe (albeit imperfectly) the speech in such spoken documents into a stream of words. But to derive content of interest, one would like to be able to apply language processing techniques that have traditionally been developed for written input.

A challenge for the processing of most classes of spoken documents - as compared with text documents - is the lack of overt segmentation information. Text input typically contains punctuation that segments words into sentences and subsentential units. Sentences are further organized into higher-level units such as speaker quotes, paragraphs, sections, chapters, articles, and so on, via formatting. In contrast, when spoken language is processed by an automatic speech recognizer, the output is simply an unannotated stream of words, as shown in the example below. Human listeners can easily segment such spoken input, arriving at the formatted version that follows. To do so they draw on a range of cues, not all of which are fully understood.

Automatic segmentation is still far from human performance, but significant progress has been made by combining lexical information from a word recognizer, with spectral and prosodic cues. Lexical sequence information provides cues related to syntactic and semantic constraints, and is thus helpful in finding sentence and clause boundaries. For example, a sentence in English is not likely to end with a determiner. Such cues are, however, subject to degradation from word recognition errors. Lexical cues can also be fairly domain-specific, and may thus perform poorly when training and test data come from different speaking contexts. Spectral information provides cues to speaker and show changes, as well as to non-speech events such as laughter. Prosodic features such as fundamental frequency, duration, and energy patterns provide information about multiple types of segment boundaries. For example, pitch tends to drop before the ends of sentences, and to an even lower value at the end of a topic or paragraph-like unit. Boundaries are often accompanied by pauses and by durational lengthening of phones directly preceding the boundary.

Unformatted Word Transcripts

with more american firepower being considered for the persian gulf defense secretary cohen to-day issued by far the administration's toughest criticism of the u. n. security council without mentioning russia or china by name cohen took dead aim at their reluctance to get tough with iraq frankly i find it uh incredibly hard to accept the proposition that in the face of saddam's uh actions that uh members of the security council cannot bring themselves to declare that this is a fundamental or material breach uh of uh conduct on his part i think it challenges the credibility of the security council in europe to-day secretary of state albright trying to gather support for tougher measures was told by the british and french ...

Formatted transcripts

Reporter: With more American firepower being considered for the Persian Gulf, defense secretary Cohen today issued by far the administration's toughest criticism of the U.N. Security Council. Without mentioning Russia or China by name, Cohen took dead aim at their reluctance to get tough with Iraq.

Cohen: Frankly I find it incredibly hard to accept the proposition that in the face of Saddam's actions that members of the Security Council cannot bring themselves to declare that this is a fundamental or material breach of conduct on his part. I think it challenges the credibility of the Security Council.

Reporter: In Europe today, Secretary of State Albright trying to gather support for tougher measures was told by the British and French ...

Over the past decade, researchers have explored methods for improving computational models for various levels of segmentation, showing that combining both acoustic and lexical cues provides significant gains in accuracy over a naive pause-based segmentation. More importantly, as will be shown here for a variety of language processing tasks, these segmentations also lead to much better task performance than pause-based segmentations. The experiments described represent a survey of the work of different groups over different time periods controlling for different aspects of segmentation, so the results are not directly comparable. However, the findings together tell a consistent story, specifically that many language processing tasks benefit from linguistic structure (beyond pause units) and that optimizing segmentation for the task is a useful strategy, i.e., the best tradeoff of recall and precision varies depending on the task.

In the remainder of this section, we describe different types of segmentation useful for spoken document processing, outline popular methods for feature extraction and computational modeling, survey recent results showing the impact of segmentation in several language processing applications, and summarize the findings and open questions.

1.4.1.2. Segmentation in Spoken Language

Language processing technology for text leverages sentence boundary and punctuation information, and the spoken language versions of this technology build on these systems. Thus, sentence segmentation is of particular importance for automatic spoken document analysis and understanding. Sentence boundaries are also important for aiding

human readability of the output of automatic speech recognition systems (Jones *et al.* 2005).

Sentence-level information is but one of many useful levels of structure in language, as evidenced by the additional forms of punctuation (for example, commas) often available in text. For some language analysis tasks, such as parsing and entity extraction, sub-sentence punctuation is of additional value. However, many of these applications may benefit more from an alternative to punctuation: prosodic phrase boundaries. Speakers naturally group words into semantically coherent phrases indicated by timing and pitch cues; these prosodic phrase boundaries often coincide with major syntactic constituent boundaries (particularly those marked with commas and semi-colons) but have a much flatter structure than syntax. They provide smaller (and potentially more useful) units for processing.

Applications of segmentation above the sentence level depend on genre. For example, topic segmentation is important when processing news broad-casts that include multiple stories. Similarly, speaker tracking and possibly role or identity recognition can provide useful structure in genres with multiple speakers. Simply knowing who is speaking (even without an associated name) can improve the read-ability of a speech transcript when there is more than one person talking. Speaker tracking is also useful for automatic analysis of conversation or meeting dynamics and for attribution in question answering. Both speaker and topic segmentation can be useful in speech recognition, for acoustic and language model adaptation, respectively.

1.4.1.3. Computational Modeling Techniques

Two very different types of segmentation algorithms are used: audio diarization and structural segmentation. Audio diarization aims to segment an audio recording into acoustically homogeneous regions, given only features extracted from the audio signal. Audio diarization techniques can include a variety of tasks, such as distinguishing speech from music or advertisements from news. The term structural segmentation is used here to include tasks that represent linguistic structure (commas, sentences, story/discourse), for which algorithms leverage both acoustic and lexical cues. The two classes of algorithms are treated separately below, followed by a discussion of how different types of segmentation may be combined. We provide an overview of the most popular methods in each case, with the disclaimer that technology is still evolving in this field.

1.4.1.4. Speaker Diarization

Much of the foundation for speaker diarization comes from speaker recognition research; some of the earliest systems were developed to support work on speaker identification in broadcast news. A typical speaker diarization system may be broken down into several “standard” components (Tranter and Reynolds 2006), with the two main components being “segmentation” and “clustering.” During the segmentation step (or “speaker change detection”), boundaries between acoustic events (typically due to a change of speaker) are located to create homogeneous segments of audio. The dominant approach to segmentation involves computing a generalized log likelihood ratio at candidate boundaries, comparing the likelihoods of the data using two distributions for the subsets of

data to the left and right of the boundary vs. a single distribution for the combined set. To determine the cut-off point, typically some form of regularization or prior is used, such as the Bayesian Information Criterion, which effectively adds a penalty for increased numbers of parameters. Then, during clustering, all of the segments belonging to the same speaker are grouped together. The most common approach for the initial speaker clustering is hierarchical agglomerative clustering, which begins with a large number of clusters that are merged pair-wise. The number of speakers is not known a priori, so various heuristics are used to determine the stopping point. Determining the number of speakers can be difficult in applications where some speak only briefly (e.g., in news sound bites), since they tend to be clustered with other speakers.

The segmentation and clustering steps may be iterated until some stopping criteria is satisfied. In subsequent passes, different models may be used, such as hidden Markov models (HMMs) for joint segmentation and clustering. Multi-pass methods are useful for the challenge of handling speaker overlap (in talk shows) and handling noisy conditions (reporters calling in from the field). The most common features used in audio diarization are cepstral features and their derivatives, as in speech recognition except without the normalization aimed at factoring out speaker and channel differences since these are exactly the types of differences that are targeted in diarization.

Speaker diarization performance is typically measured by diarization error rate (DER), which measures the percentage of time that a system incorrectly labels the audio recording based on an automatic mapping of hypothesized speaker clusters to reference speakers according to maximal overlap.¹⁸ Diarization error rates vary depending on the number of speakers, speaker overlap, and acoustic conditions. Hence, DER can vary widely, e.g., from roughly 2% to 12% for different broadcast news sources (Tranter and Reynolds 2006).

1.4.1.5. Structural Segmentation

There are two basic modeling approaches used for structural segmentation: 1) detection of boundary events, and 2) whole constituent modeling. The approaches can also be combined. Both models are applied after speech recognition, and take advantage of the alignment between words (and the phones therein) and the acoustic speech signal.

Boundary event detection is essentially a sequence tagging problem: for each word in the sequence, assign a boundary label indicating the constituent (or none) ending at that point. As such, any computational model for tagging is applicable here. HMM-like models dominated early work in speech segmentation (Stolcke and Shriberg 1996). Given the word sequence W and the prosodic features F , the most likely event sequence E is given by:

$$\hat{E} = \operatorname{argmax}_E P(E|F, W) \approx \operatorname{argmax}_E P(W, E) P(F|E)$$

The transition probabilities (in $P(W, E)$) are obtained from an n -gram language model, also referred to as a hidden-event language model, that characterizes the event labels

¹⁸ <http://www.nist.gov/speech/tests/rt/rt2007/>

and words jointly. The observation posteriors $P(F|E)$ are generated from a prosody model, e.g., a decision tree classifier or neural network. HMMs that are discriminatively trained have also been used for sentence boundary detection (Tomalin and Woodland 2006).

Maximum entropy (Maxent) and conditional random field (CRF) classifiers have also been investigated for boundary event detection (Huang and Zweig 2002; Liu *et al.* 2006). Unlike HMMs, Maxent and CRF approaches provide more freedom to incorporate contextual information and to combine word-based and prosodic features. Both use the exponential form for the conditional probabilities. For example, in Maxent:

$$P(E_i|W, F) = \frac{1}{Z_\lambda(W, F)} \exp \left(\sum_k \lambda_k g_k(E_i, W, F) \right)$$

A CRF models sequence information, whereas Maxent individually classifies each data sample. The weights (λ) for the features are estimated to maximize the conditional probabilities of the training set. Liu *et al.* (2006) compare HMM, Maxent, and CRF approaches for sentence segmentation of broadcast news and conversational speech, finding that the CRF leads to the best results but by a small margin and at a higher computational cost. Another approach that can accommodate a rich variety of features is based on combining Boostexter with a hidden-event language model (Zimmermann *et al.* 2006). Boostexter is based on the principle of boosting that combines many weak classifiers, each having a basic form of one level decision trees using confidence-rated prediction. It has the advantage of good performance with a relatively low cost implementation.

Whole constituent modeling considers both the beginning and the end time of a segment in determining boundary location. For many problems, the cues are local to the boundary, such as for prosodic phrase boundaries. For others, the cues extend over the entire phrase, and the whole constituent approach is preferable. Whole constituent modeling is also useful when a maximum or minimum length constraint is needed. The challenge of modeling the full constituent is in decoding. Since it is impractical to consider all possible constituent onset times, the search space is often reduced by restricting the set of candidate boundaries. Whole constituent modeling has been used for sentence segmentation, story segmentation, and in speaker modeling where both acoustic and lexical cues are incorporated. In sentence segmentation for translation (Matusov *et al.* 2006), an explicit sentence length model is incorporated in a log-linear combination of language model and prosody model scores. Posterior probabilities identified via boundary event detection can be included in the combination for further improvements (Matusov *et al.* 2007). In story segmentation, whole constituent modeling is needed for characterizing the topical coherence of sentences in the segment and extracting position-based information about lexical cue words (Rosenberg *et al.* 2007).

The modeling approaches described above rely on various word-based and prosodic features. Lexical features typically consist of word n -grams and part-of-speech n -grams. These features are useful for identifying short utterances in spontaneous speech such as backchannels (“uhhuh”, “yeah”), for characterizing sequences of words that are unlikely to be split by a sentence boundary (“the problem”), and for representing words that are

likely to start a new sentence (such as “I”). Syntactic features have also been used to improve sentence boundary detection (Roark *et al.* 2006a; Favre *et al.* 2008). The features have different representations in different modeling approaches, for example, an n -gram language model in the HMM framework or word tuple indicators in discriminative classifier approaches.

Prosodic features reflect information about duration, pause, intonational and energy contours. Features can be extracted from automatic alignments of word and phone transcriptions with the speech signal. Duration features (such as word, pause, and phone durations) are obtained directly from alignment time marks. Since different phones have different baseline durations, duration features are typically normalized for phonetic content. In addition they may be normalized by speaker or speaking rate. Pause duration is known to be important for segmentation, but it poses some challenges because of editing in broadcast news and speaker overlap in conversational speech. Useful pitch and energy features tend to capture differences across the word boundary in question, as well as slopes and normalized level of pitch or energy just before a boundary. In the case of both pitch and energy, features must be appropriately normalized (by speaker for pitch; by channel for energy).

Sentence segmentation is reported either using an F-measure or the sentence error rate, which computes error rate as the number of incorrect boundaries over the total number of boundaries using the mdeval tool.¹⁹ Performance depends on speech genre and recognizer error rate. Conversational genres tend to have shorter sentences but higher error rates. With the caveat that performance continues to improve, example F-measures on automatically transcribed speech are roughly 65% for broadcast news (Favre *et al.* 2008a) and 64% for broadcast conversations in English. The F for Mandarin broadcast news is about 75% (Matusov *et al.* 2007).

1.4.1.6. Multi-level Segmentation

Since the various types of segmentation are generally interdependent and since automatically detected boundaries can be errorful, soft predictions (boundary posteriors) at the different levels can be considered jointly to improve performance. Speaker boundaries based purely on acoustic information often do not align perfectly with sentence boundaries that are based on speech recognizer output. Higher accuracy speaker boundaries can be obtained by adjusting boundary times to match those of nearby sentence boundaries. Similarly, improved story boundary detection is achieved by considering candidate boundary points at more locations than the automatically detected sentence boundaries, either by lowering the threshold for sentence detection (e.g. from probability 0.5 to probability 0.1) or simply by considering all boundaries with a 250ms or greater length pause (Rosenberg *et al.* 2007). Taking into consideration the higher-level information associated with story boundary detection can potentially feedback into improvements in sentence segmentation. The use of soft decisions on segment boundaries also makes it possible to tune the boundary detection threshold or operating point for specific applications. Work described in the next section shows that this is indeed useful, though the best operating point varies with the different tasks.

¹⁹ <http://www.nist.gov/speech/tests/rt/2004-fall/>

1.4.1.7. Applications

Spoken document processing can involve a combination of several tasks, typically starting with speech recognition and speaker segmentation, followed by some basic linguistic analysis such as part-of-speech tagging and parsing, and then involving higher level processing such as translation and information extraction. Automatic segmentation touches on all of these problems, but we will focus on stages after speech recognition. In the examples here, the segmentation types used (speaker, sentence, comma, intonational phrase, and story) employ the basic algorithms described in Section 1.4.1.3. The specific methods vary with genre and with the time period of the work, since this is still an area of research and the best case configurations are evolving.

1.4.1.8. Speaker Role and Identity Recognition

In broadcast news, most speech is from anchors and reporters, but there are excerpts from speeches or interviews, sometimes referred to as “soundbites.” Detecting soundbites and associating them with particular speakers is important for information extraction and attribution in question answering. Using the example in Section 1.4.1.1, the task is to associate the speech segments produced by Cohen with his name, given speaker diarization results and ASR transcripts.

Data from Mandarin broadcast news has been used for soundbite and speaker name recognition (Liu and Liu 2007) using a classification framework. Each speaker turn is labeled with one of three roles: anchor, reporter, or soundbite. The features used are based on textual information (mainly word n-grams) from the current segment, the preceding and the following segments. Speaker name recognition takes advantage of the coded behavior typical of broadcast news (i.e., reporters often naming the next or previous speaker). Hypothesized names from the current and neighboring segments are classified in terms of whether or not it is the speaker’s name for a target soundbite segment using keywords and position of the name in the sentence.

In experiments with Mandarin broadcast news, word errors and sentence segmentation errors have different impacts on the soundbite detection and name recognition tasks due to the types of cues they use. For soundbite detection, segmentation errors were more harmful than word errors, with degradation in F-measure of 12% vs. 2%, respectively, compared to detection on oracle transcripts. Incorrect sentence segmentation leads to missed cue words to soundbites. For name recognition, the opposite is true: 1% degradation of F-measure due to segmentation errors, compared to 15% with word errors. Since many soundbite speaker names are infrequent, they are less reliably recognized than other words.

1.4.1.9. Tagging and Parsing

Part-of-speech (POS) tagging is the process of marking up a sequence of words with their parts of speech (e.g., noun, verb). Parsing produces a structural analysis of a word sequence with respect to a grammar. High quality automatic sentence segmentation is important for utilizing these techniques most effectively, both for accuracy and for addressing length-dependent complexity issues. Experiments on the impact of comma pre-

diction on POS tagging accuracy of Mandarin broadcast news speech showed that performance was significantly better when using the automatic commas compared to a matched train/test condition without sentence-internal punctuation (Hillard *et al.* 2006). Other work on parsing English conversational speech showed a significant effect of sentence segmentation on parsing whether using the reference word transcriptions (Kahn *et al.* 2004) or ASR transcripts (Harper *et al.* 2005): automatic sentence segmentation can recover roughly half the loss in parsing performance due to using pause-based segmentation compared to hand-labeled references (roughly 10 points F-measure recovered).

Further experiments (Roark *et al.* 2006a; Harper *et al.* 2005) showed that optimizing sentence segmentation thresholds specifically for parse accuracy (vs. sentence segmentation accuracy) yielded greater improvements in parsing (from F of 64.8 to 65.7). When optimizing for parse accuracy, the system tended to produce shorter segments than when optimizing for sentence segmentation accuracy, i.e. trading off precision for recall. The shorter sentence-like segments also benefited a parsing language model used in speech recognition, leading to significant improvements in the SParseval score when word sequences and parses are chosen jointly.

1.4.1.10. Information Extraction

Information Extraction (IE) aims at finding semantically defined entities in documents and characterizing relations between them. Like many text processing tasks, IE systems typically benefit from handwritten punctuation. Studies show that when punctuation is removed, there is an associated loss in performance. For example, missing commas have a dramatic impact on IE (Makhoul *et al.* 2005), with performance loss typically bigger than that observed when moving from reference to ASR output (for a range of word error rates on English news). Similar results were reported for name tagging on Mandarin broadcast news (Hillard *et al.* 2006), and it was shown that half of the lost performance was recovered with automatic comma prediction (from F of 84.9 to 85.4).

Another study (Favre *et al.* 2008a) confirmed these observations for English IE on speech, and found that optimizing sentence and comma prediction thresholds for IE performance is more effective than optimizing these thresholds separately for punctuation prediction accuracy: improvements in the ACE scores are from 15.6 to 18.4 for relations, and 47.0 to 48.2 for entities. Error analysis showed that punctuation errors can result in merged noun phrases or split entities. The best case performance was obtained by jointly optimizing comma and sentence boundary thresholds, but allowing the thresholds to vary in detecting entities vs. relations.

1.4.1.11. Machine Translation

In machine translation (MT), sentence segmentation helps provide translations with proper punctuation, but it also impacts word choice since sentence boundaries are incorporated in the language model and they constrain the possible phrase translations. Many systems limit sentence lengths for complexity reasons, which motivates a constituent-based approach to sentence segmentation. The translation application also motivates a new type of feature, introduced by Matusov *et al.* (2007) to characterize phrase coverage of the words that span the candidate boundaries to ensure that word sequences with good

phrasal translations will not be broken by a segment boundary. The phrase coverage feature is a bigram language model probability. Depending on whether the bigram probability is high or low, there is likely to be a good phrasal translation in the system or not, respectively.

Different sentence segmentation algorithms have been evaluated on large vocabulary Arabic-to-English and Chinese-to-English broadcast news translation tasks using the phrase-based MT system of RWTH (Mauser *et al.* 2006). The explicit length modeling of the whole constituent model (using a less sophisticated prosody model and without the phrase coverage feature) did not do as well as the boundary detection approach in terms of sentence segmentation accuracy, but it did lead to better MT performance. MT performance improves by combining the two methods, but the best result was achieved by using the phrase coverage feature, increasing BLEU from 18.1 for fixed-length segments to 21.2 for the MT-optimized sentence predictions. The sentence boundary precision is reduced significantly when the phrase coverage feature is used, but this does not affect the translation because the context at the erroneously inserted boundaries was not captured in MT training anyway. As in the parsing work, MT experiments have shown that a lower detection threshold is better for translation of Chinese (0.2 vs. the minimum error threshold of 0.5), favoring recall over precision or shorter segments. A separate study on Arabic-to-English translation found that longer sentences are better (Matsoukas *et al.* 2007). Shorter sentences in Chinese are likely to help limit reordering errors, while for Arabic (which has less long distance reordering), longer segments likely provide additional context without much increased risk of reordering mistakes.

While punctuation marks predicted in ASR output can be useful for predicting target language punctuation, they can be also used to guide the MT process. Matusov *et al.* (2007) used automatically predicted Chinese commas as soft boundaries for reordering in MT search. Reordering across a comma is found to be highly unlikely and is penalized by modifying the lexicalized reordering model of the phrase-based MT system (Zens and Ney 2006), with reduced penalties for lower confidence predictions. In experiments on the Chinese-to-English task, the soft comma constraints did not result in a significant improvement in standard MT development scoring methods (BLEU and TER). However, the word order in several translated sentences was subjectively better when the soft boundary penalty was applied.

1.4.1.12. Conclusions

In summary, the fact that most language technology used in spoken document processing is designed in large part from written text argues that speech must be made to look more like text for achieving good performance. An important challenge in this respect is speech segmentation, including sentence segmentation at a minimum, but ideally also speaker and topic segmentation for formatting and adaptation, as well as sub-sentence punctuation and/or intonational phrase prediction. Both event-based and constituent-level computational models have been developed for these problems, many of which combine lexical and acoustic cues in detecting boundaries. While these algorithms are far from perfect, in most applications they provide a much better solution than simple pause-based segmentation. Oracle results suggest that further improvements to segmentation algorithms would be useful, though improvements to word recognition have higher impact for some tasks.

In the various applications surveyed here, there is a consistent finding that tuning the segmentation thresholds for the application leads to significant performance improvements over using the threshold that minimizes segmentation error. In many cases (but not all), higher recall is more effective (i.e. shorter sentences), but the optimal threshold varies. This raises the question as to how best to meet the needs of multiple language processing modules, particularly when they all operate on the same hypothesized transcript. One solution is to use a low threshold (more hypothesized boundaries) with confidences associated with the boundaries, so that different downstream modules can use their own threshold. Alternatively (or in addition), the need for different thresholds may reflect a need for different types of structures, including subsentence units such as intonational phrases or syntactic chunks.

1.4.2. CADIM Arabic Tools: Morphological Analysis, Disambiguation and Generation, Tokenization, Diacritization, Lemmatization, POS Tagging and Base Phrase Chunking

Authors: Mona T. Diab, Nizar Habash, Owen Rambow, and Ryan M. Roth

1.4.2.1. Introduction

In this section, we describe some of the NLP tools we developed over the past number of years to process Modern Standard Arabic and Arabic Dialect text. We present a series of tools that vary in their dependence on explicit linguistic rules and linguistic depth, and that target a variety of NLP tasks. AL-MORGEANA and MAGEAD are morphological analysis and generation systems. AL-MORGEANA extends on the Buckwalter analyzer and MAGEAD handles Arabic dialects. MADA is a system for morphological disambiguation including POS tagging, lemmatization and diacritization. TOKAN is a general tokenizer for Arabic that works with MADA. AMIRA is a suite of shallow linguistic processing tools with limited dependence on explicit linguistic rules; AMIRA performs tokenization, POS tagging, and base phrase chunking. We compare some of the functionality of our different systems. We also present several of the NLP applications that have successfully used our different systems.

Morphological Analysis refers to the process by which a word (defined orthographically for our purposes) has all of its possible morphological analyses determined. Each analysis also includes a single choice of core part-of-speech (such as noun or verb; the exact set is a matter of choice). A morphological analysis can be either form-based, in which case we divide word into all of its constituent morphemes, or functional, in which case we also interpret these morphemes. For example, in broken (i.e., irregular) plurals, a form-based analysis may not identify the fact that the word is a plural, since it lacks the usual plural morpheme, while functional analysis would.

Morphological disambiguation refers to the choice of a morphological analysis in context. This task for English is referred to as part-of-speech (POS) tagging, since the standard POS tagset, though only comprising 46 tags, completely disambiguates English morphologically. In Arabic, the corresponding tagset comprises up to 330,000 theoretically possible tags, so the task is much harder. Reduced tagsets have been proposed for Arabic as well, in which certain morphological differences are conflated, making the morphological disambiguation task easier. The term POS tagging is usually used for Arabic with respect to some of the smaller tagsets.

Tokenization (also sometimes called *segmentation*) refers to the division of a word into clusters of consecutive morphemes, one of which typically corresponds to the word stem, usually including inflectional morphemes. There are two parameters. First, we need to choose which types of morphemes to segment and how to present them (the tokenization scheme). There is no single correct tokenization scheme. Second, we need to decide whether after removing some morphemes, we regularize the orthography of the resulting segments, since the concatenation of morphemes can lead to spelling changes on their boundaries. For example, the Ta-Marbuta (ﻻ *p*) appears as a regular Ta (ﺕ *t*) when followed by a pronominal clitic.²⁰ Usually, the term *segmentation* is only used when no orthography regularization takes place. Orthography regularization is desirable in NLP because it reduces data sparseness, as does tokenization itself.

Lemmatization is the mapping of a word form to its corresponding lemma (also known as a *citation form*), the canonical representative of its lexeme. A lexeme is a lexicographic abstraction: it is the set of all word forms that differ only in inflection and possible cliticization, processes which are (almost) always productive. The lemma is a conventionalized choice of one of these word forms to stand for the set. Usually, a lexeme is assumed to have one sense, so that homonyms (such as the multiple meanings of English word ‘bank’) are considered two lexemes. If the sense distinction is removed from the definition of lexemes, one sometimes uses the term *vocable*, which is a purely morphological characterization of a set of word forms. In this chapter, we use the term lexeme without the semantic dimension, i.e., we collapse the concepts of lexeme and vocable.

Diacritization is the process of adding missing diacritics (short vowels, the marker of the absence of a short vowel, and the gemination marker). Diacritization is closely related to morphological disambiguation and to lemmatization: for an undiacritized word form, different morphological analyses often correspond to different diacritizations (for example, voice in MSA), and different lemmas can be seen to lead to different diacritizations as well. Often, the choice of the diacritic on the last written letter of the word (without the possessive or object clitic which may be attached) is particularly hard, since it requires syntactic information: in verbs, this diacritic often expresses mood, and in nouns and adjectives, it expresses syntactic case. Thus, it is often common to define a simpler diacritization task which does not choose the word final diacritic.

²⁰ Arabic romanization is in the Buckwalter transliteration scheme (Buckwalter, 2002).

Base phrase chunking is a syntactic task, in which non-recursive multiword syntactic phrases are identified without actual parsing. The relevant notion of phrase needs to be clearly defined depending on where and how these syntactic chunks can be used, and in Arabic, this is not as straightforward as in English due to the complexity of some of the syntactic structures of noun phrases.

The Columbia Arabic and its Dialects Modeling (CADIM) group has produced a set of tools to address these tasks. In this chapter we will present the functionality of several of our basic tools that have been widely used in the NLP community and in our own research. This chapter is a survey chapter, it is meant to highlight the various tools that we have and how they compare with one another. We will also refer to some of the applications where we used our tools.

Our tools can be classified coarsely into a set of deep and shallow processing tools as they vary in their dependence on explicit linguistic representations in their machinery. The deep processing tools depend on explicitly expressed linguistic analyses as an essential component in the processing of the text, and require corpora that have been annotated at this level. The shallow processing tools, on the other hand, learn from surface representations where the morphological information and linguistic knowledge is implicitly represented and the system learns pattern generalizations without reference to explicitly represented linguistic units.

1.4.2.2. ALMORGEANA

ALMORGEANA is a system for Arabic Lexeme-based Morphological Generation and Analysis (Habash 2007).²¹

Desiderata: We chose the following desiderata for the design of ALMORGEANA (and also MAGEAD, Section 1.4.2.3): (1) coverage of the language of interest in terms of both lexical coverage (large scale) and coverage of morphological and orthographic phenomena (robustness); (2) a complete mapping between surface word forms to/from a functional representation of morphology; (3) full reversibility of the system so it can be used as an analyzer or a generator; (4) usability in a wide range of natural language processing applications such as MT or IR; and finally, (5) availability for the research community.

Lexicon: ALMORGEANA is built on top of the publicly available large-scale database of the Buckwalter Arabic Morphological Analysis (BAMA) (Buckwalter 2002; Buckwalter 2004). The database consists of six components: three morphological databases for prefixes, stems, and suffixes; and three tables showing the compatibility of entries from the three morphological databases with each other (prefix-stem, stem-suffix and prefix-suffix). Unlike BAMA, which focuses on analysis to a surface form-based representation, ALMORGEANA analyzes to, and generates from the functional (lexeme-and-feature) level of representation. To that effect, the ALMORGEANA lexicon extends the BAMA morphological databases with lexeme and feature keys, which are used in

²¹ A previous publication about ALMORGEANA focused on the generation component of the system which was named Aragen (Habash, 2004).

analysis and generation. This work on ALMORGEANA is close in spirit to the extensions to BAMA in the functional morphology system, ELIXIRFM (Smrž 2007).

Analysis: Analysis in ALMORGEANA is similar to BAMA: the word is segmented into prefix-stem-suffix triples, whose individual presence and bi-lateral compatibility is checked against the BAMA database. The difference lies in an extra step that uses lexeme and feature keys associated with stem, prefix and suffix string sequences to construct the lexeme and feature output. The output of ALMORGEANA includes all possible analyses. Each analysis consists of the diacritized form of the word, its lexeme, its morphological features, and an English gloss. For example the word للكتب *llktb* ‘for the books’ returns the following analysis:

`lilkutubi=[kitAb_1 POS:N l+ Al+ +PL +GEN]=books`

Here, *lilkutubi* is the diacritized form of the word. Inside the square brackets, we find the nominal lexeme *kitAb_1* ‘book’, the proclitic preposition *l+* ‘to/for’, the definite article *Al+* ‘the’, the feature *+PL* ‘plural’ and the feature *+GEN* ‘genitive case’. Most of the information in the feature set is directly derivable from the morpheme tags in the BAMA output for the same word: `li/PREP+Al/DET+kutub/NOUN+i/CASE_DEF_GEN`. However the feature *+PL* indicating plurality is not. It is part of the extension done in ALMORGEANA in processing the BAMA databases.

The challenge of out-of-lexicon words during analysis is handled in two ways similar to BAMA. First, simple spelling expansion is used for cases with common spelling variations such as Alef-Hamza- Above (أ >) spelled without the Hamza (ء ’) as Alef (أ A), and Ya (ي y) spelled without the dots as Alef-Maqsura (آ Y). Second, a back-off analysis mode can be turned on in which prefix and suffix matching and compatibility is used but untested stems can be hypothesized. A full description is given by Habash and Rambow (2005).

Generation: In generation, the input is a lexeme and feature set. The generated output is a fully inflected and diacritized word. For example, `[kitAb_1 POS:N l+ Al+ +PL +GEN]` generates *lilkutubi*. The process of generating from lexeme and features is similar to analysis except that lexeme and feature keys are used instead of string sequences. First, the feature set is expanded to include all forms of under-specified obligatory features, such as case, gender, number, etc. Next, all lexeme and feature keys in the ALMORGEANA lexicon that fully match any subset of the lexeme and expanded feature set are selected. All combinations of keys that completely cover the lexeme and expanded feature set are matched up in prefix-stem-suffix triples. Then, each key is converted to its corresponding prefix, stem or suffix string. The same compatibility tables used in analysis are used to accept or reject prefix-stem-suffix triples. Finally, all unique accepted triples are concatenated and output. In the case that no surface form is found, a back-off solution that attempts to regenerate after discarding one of the input features is explored.

See (Habash 2007) for more details on ALMORGEANA and an evaluation of its performance.

1.4.2.3. MAGEAD

MAGEAD is a morphological analyzer and generator for the Arabic language family, by which we mean both MSA and the spoken dialects. We shall collectively refer to MSA and its dialects as Arabic variants. For a fuller discussion of MAGEAD (including an evaluation), see (Habash *et al.* 2005; Habash and Rambow 2006). For an excellent discussion of related work, see (Al-Sughaiyer and Al-Kharashi 2004).

MAGEAD relates (bidirectionally) a lexeme and a set of linguistic features to a surface word form through a sequence of transformations. In a generation perspective, the features are translated to abstract morphemes which are then ordered, and expressed as concrete morphemes. The concrete templatic morphemes are interdigitated and affixes added, and finally morphological and phonological rewrite rules are applied. In this section, we discuss our organization of linguistic knowledge, and give some examples; a more complete discussion of the organization of linguistic knowledge in MAGEAD is presented by Habash *et al.* (2006).

Lexeme and Features: Morphological analyses are represented in terms of a lexeme and features. We define the *lexeme* to be a triple consisting of a root, a *morphological behavior class* (MBC), and a meaning index. We do not deal with issues relating to word sense here and therefore do not further discuss the meaning index. It is through this view of the lexeme (which incorporates productive derivational morphology without making claims about semantic predictability) that we can both have a lexeme-based representation, and operate without a lexicon (as we may need to do when dealing with a dialect). In fact, because lexemes have internal structure, we can hypothesize lexemes on the fly without having to make wild guesses (we know the pattern, it is only the root that we are guessing). Our evaluation shows that this approach does not wildly over-generate.

We use as our example the surface form, ازدهرت *Aizdaharat* (*Azdhrt* without diacritics) ‘she/it flourished’. The MAGEAD lexeme-and-features representation of this word form is as follows:

(1) Root:zhr MBC:verb-VIII POS:V PER:3 GEN:F NUM:SG ASPECT:PERF

Morphological Behavior Class: An MBC maps sets of linguistic feature-value pairs to sets of abstract morphemes. For example, MBC verb-VIII maps the feature-value pair ASPECT:PERF to the abstract root morpheme [PAT_PV:VIII], which in MSA corresponds to the concrete root morpheme *V1tV2V3*, while the MBC verb-II maps ASPECT:PERF to the abstract root morpheme [PAT_PV:II], which in MSA corresponds to the concrete root morpheme *IV22V3*. We define MBCs using a hierarchical representation with non-monotonic inheritance. The hierarchy allows us to specify only once those feature-to-morpheme mappings for all MBCs which share them. For example, the root node of our MBC hierarchy is a word, and all Arabic words share certain mappings, such as that from the linguistic feature conj:w to the clitic w+. This means that all Arabic words can take a cliticized conjunction. Similarly, the object pronominal clitics are the same for all transitive verbs, no matter what their templatic pattern is. We have developed a specification language for expressing MBC hierarchies in a concise manner. Our hypo-

thesis is that the MBC hierarchy is variant-independent, i.e. dialect/MSA independent. However, as more variants are added, some modifications may be needed. Our current MBC hierarchy specification for both MSA and Levantine, which covers only the verbs, comprises 66 classes, of which 25 are abstract, i.e., only used for organizing the inheritance hierarchy and never instantiated in a lexeme.

MAGEAD Morphemes: To keep the MBC hierarchy variant-independent, we have also chosen a variant-independent representation of the morphemes that the MBC hierarchy maps to. We refer to these morphemes as *abstract morphemes* (AMs). The AMs are then ordered into the surface order of the corresponding concrete morphemes. The ordering of AMs is specified in a variant-independent context-free grammar. At this point, our example (1) looks like this:

[Root:zhr][PAT_PV:VIII] [VOC_PV:VIII-act] + [SUBJUF_PV:3FS]

Note that as the root, pattern, and vocalism are not ordered with respect to each other, they are simply juxtaposed. The ‘+’ sign indicates the ordering of affixival morphemes. Only now are the AMs translated to *concrete morphemes* (CMs), which are concatenated in the specified order. Our example becomes:

(3) <zhr,V1tV2V3,iaa> +at

Simple interdigitation of root, pattern and vocal-ism then yields the form iztahar+at.

MAGEAD Rules: We have two types of rules. *Morphophonemic/phonological* rules map from the morphemic representation to the phonological and orthographic representations. For MSA, we have 69 rules of this type. Orthographic rules rewrite only the orthographic representation. These include, for examples, rules for using the *shadda* (consonant doubling diacritic). For Levantine, we have 53 such rules.

For our example, we get /izdaharat/ at the phonological level. Using standard MSA diacritized orthography, our example becomes Aizdaharat (in transliteration). Removing the diacritics turns this into the more familiar ازدهرت *Azdhrt*. Note that in analysis mode, we hypothesize all possible diacritics (a finite number, even in combination) and perform the analysis on the resulting multi-path automaton.

We follow (Kiraz 2000) in using a multi-tape representation. We extend the analysis of Kiraz by introducing a fifth tier. The five tiers are used as follows: Tier 1: pattern and affixational morphemes; Tier 2: root; Tier 3: vocalism; Tier 4: phonological representation; Tier 5: orthographic representation. In the generation direction, tiers 1 through 3 are always input tiers. Tier 4 is first an output tier, and subsequently an input tier. Tier 5 is always an output tier.

We have implemented multi-tape finite state automata as a layer on top of the AT&T two-tape finite state transducers (Mohri *et al.* 1998). We have defined a specification language for the higher multi-tape level, the new MORPHTOOLS format. Specification in the MORPHTOOLS format of different types of information such as rules or context-free

grammars for morpheme ordering are compiled to the appropriate LEXTOOLS format (an NLP-oriented extension of the AT&T toolkit for finite-state machines, (Sproat 1995)). For reasons of space, we omit a further discussion of MORPHTOOLS. For details, see (Habash *et al.* 2005).

From MSA to Levantine: We have modified MAGEAD so that it accepts Levantine rather than MSA verbs. Our effort concentrated on the orthographic representation; to simplify our task, we used a diacritic-free orthography for Levantine developed at the Linguistic Data Consortium (Maamouri *et al.* 2006). Changes were done only to the representations of linguistic knowledge, not to the processing engine. We modified the MBC hierarchy, but only minor changes were needed. The AM ordering can be read off from examples in a fairly straightforward manner; the introduction of an indirect object AM would, for example, require an extension of the ordering specification. The mapping from AMs to CMs, which is variant-specific, can be obtained easily from a linguistically trained (near-)native speaker or from a grammar handbook. Finally, the rules, which again can be variant-specific, require either a good morphophonological treatise for the dialect, a linguistically trained (near-)native speaker, or extensive access to an informant. In our case, the entire conversion from MSA to Levantine was performed by a native speaker linguist in about six hours.

1.4.2.4. MADA

MADA (Morphological Analysis and Disambiguation for Arabic) is a utility that, given raw Arabic text, adds as much lexical and morphological information as possible by disambiguating in one operation part-of-speech tags, lexemes, diacritizations and full morphological analyses. This task is not trivial, due to the morphological complexity of the language. The complexity of the morphology together with the underspecification of the orthography creates a high degree of ambiguity. On average, a word form in the Penn Arabic Treebank (Maamouri *et al.* 2004) has about 12 morphological analyses.

MADA's approach distinguishes between the problems of morphological analysis (what are all the different readings of a word without regard to context) which is handled by a morphological analyzer such as ALMORGEANA or MAGEAD, and morphological disambiguation (what is the correct reading in a specific context). MADA is a morphological disambiguation system. Once a morphological analysis is chosen in context, we can determine its full POS tag, lemma and diacritization in a single step. Knowing the morphological analysis also allows us to tokenize and stem deterministically; this is handled by TOKAN (Section 1.4.2.5) once MADA has finished processing the text.

MADA operates in stages. First, it uses ALMORGEANA (Section 1.4.2.2) internally to produce a list of potential analyses for each word encountered in the text; at this point, word context is not considered. MADA then makes use of up to 19 features to rank the list of analyses. For each feature, a classifier is used to create a prediction for the value of that feature for each word in its context. Fourteen of the features use Support Vector Machine (SVM) classifiers; the remaining features capture information such as spelling variations and n-gram statistics.

Each classifier prediction is weighted using a tuning set, and the collection of feature predictions is compared to the list of potential morphological analyses. Those analyses

that more closely agree with the weighted set of feature predictions receive higher-ranking scores than those, which do not; the highest scoring analysis is flagged as the correct analysis for that word in that context.

Once MADA has finished, the user has access to the full morphological analysis for each word in the input. The user may then extract any or all of the analysis information. This is why MADA can be used for a multitude of tasks, including part-of-speech and morphological feature tagging, lemmatization, predicting full diacritization, glossing, stemming and others. Since MADA selects a complete analysis from ALMORGEANA, all decisions regarding morphological ambiguity, lexical ambiguity, tokenization, diacritization and POS tagging in any possible POS tag set are made in one fell swoop (Habash and Rambow 2005; Habash and Rambow 2007; Roth *et al.* 2008). MADA has over 96% accuracy on basic morphological choice (including tokenization but excluding syntactic case, mood, and nunation) and on lemmatization. MADA has over 86% accuracy in predicting full diacritization (including syntactic case and mood). Detailed comparative evaluations are provided in the following publications: (Habash and Rambow 2005; Habash and Rambow 2007; Roth *et al.* 2008).

The operation of MADA is versatile and highly configurable. Starting with version 2.0, MADA applies weights to each of the 19 features it uses for better accuracy; these weights are determined on a tuning set and are optimized for different purposes, such as tokenization, diacritization, or POS tagging. These weight sets are included with the package and should be chosen by the user depending on how MADA will be used. However, users can also choose to set these weights directly themselves. By default, MADA attempts to rank complete analyses in terms of overall correctness. By choosing an alternative feature and weight set, it is possible to have MADA focus more specifically on getting a particular analysis aspect correct. For example, users can achieve a 0.4% absolute improvement in POS tagging accuracy if they use the weight set that was tuned for POS tagging, as opposed to the default set. However, the accuracy of the other MADA outputs (the lexeme prediction, for example) may suffer. Starting with version 2.1, MADA also includes a morphological back off procedure, which can be turned on or off by the user (see Section 1.4.2.2).

1.4.2.5. TOKAN

TOKAN is a general tokenizer for Arabic that provides an easy-to-use resource for tokenizing MADA disambiguated Arabic text into a large set of possibilities (Habash and Sadat 2006; Habash 2007). The decision on whether an Arabic word has a conjunction or preposition clitic is made in MADA; but the actual tokenization of the clitics including handling various morphotactics and spelling regularization is done in TOKAN. The tokenization scheme can be used as parameter in machine learning for a variety of applications, such as machine translation, or named-entity recognition (see Section 1.4.2.8.).

وسينهي الرئيس جولته بزيارته الى تركيا						
Input	<i>wsynhY</i>	<i>Alr}ys</i>	<i>jwlth</i>	<i>bzyArp</i>	<i>AlY</i>	<i>trkyA</i>
Gloss	and will finish	the president	tour his	with visit	to	Turkey
English	The president will finish his tour with a visit to Turkey.					
Scheme						
ON	<i>wsynhy</i>	<i>Alr}ys</i>	<i>jwlth</i>	<i>bzyArp</i>	</Y	<i>trkyA</i>
D1	<i>w+ synhy</i>	<i>Alr}ys</i>	<i>jwlth</i>	<i>bzyArp</i>	</Y	<i>trkyA</i>
D2	<i>w+ s+ ynhy</i>	<i>Alr}ys</i>	<i>jwlth</i>	<i>b+ zyArp</i>	</Y	<i>trkyA</i>
D3	<i>w+ s+ ynhy</i>	<i>Al+ r}ys</i>	<i>jwlp +h</i>	<i>b+ zyArp</i>	</Y	<i>trkyA</i>
WA	<i>w+ synhy</i>	<i>Alr}ys</i>	<i>jwlth</i>	<i>bzyArp</i>	</Y	<i>trkyA</i>
TB1	<i>w+ synhy</i>	<i>Alr}ys</i>	<i>jwlp +h</i>	<i>b+ zyArp</i>	</Y	<i>trkyA</i>
TB2	<i>w+ s+ ynhy</i>	<i>Alr}ys</i>	<i>jwlp +h</i>	<i>b+ zyArp</i>	</Y	<i>trkyA</i>
MR	<i>w+ s+ y+ nhy</i>	<i>Al+ r}ys</i>	<i>jwl +p +h</i>	<i>b+ zyAr +p</i>	</Y	<i>trkyA</i>
LEM	<i>>nhy</i>	<i>r}ys</i>	<i>jwlp</i>	<i>zyArp</i>	</Y	<i>trkyA</i>
POS1	V	N	N	N	P	PN
POS2	VBP	NN	NN	NN	IN	NNP
ENX	<i>w+ s+ >nhy_{VBP}</i> <i>+S3MS</i>	<i>Al+ r}ys_{NN}</i>	<i>jwlp_{NN} +h</i>	<i>b+ zyArp_{NN}</i>	</Y _{IN}	<i>trkyA_{NNP}</i>
DIAC	<i>wasayunohiy</i>	<i>Alra}iyosu</i>	<i>jawolatahu</i>	<i>biziyArapK</i>	</laY	<i>turokiyA</i>

Table 1.14: Some of the preprocessing schemes supported by TOKAN: **ON** (orthographic normalization), **D1**, **D2** and **D3** (different degrees of decliticization), **WA** (wa+ decliticization), **TB1** and **TB2** (old and new Arabic Treebank tokenization, respectively), **MR** (morphemes), **LEM** (lemmatization), **POS1** and **POS2** (two possible POS tag sets), **ENX** (a tokenization equivalent to D3+LEM+POS2 with markers for verbal subject) and **DIAC** (diacritization).

TOKAN takes as input a MADA-disambiguated file and a tokenization scheme description that specifies tokenization target. Consider the following specification:

“w+ f+ b+ k+ l+ s+ Al+ REST + / + POS +P: +O: -DIAC”

This scheme separates conjunctions, prepositions, verbal particles, the definite article and pronominal clitics and it adds the basic POS tag to the form of the word. The scheme also specifies that diacritics be generated. An analysis of the word *وسيكاتبها* *wasayukAtibuhA* ‘and he will correspond with her’ would be tokenized as “wa+ sa+yukAtibu/V +hA.” A simpler scheme such as “w+ f+ REST” would simply produce “w+ sykAtbhA.” See (Habash and Sadat, 2006) for a detailed description of several schemes that have become commonly followed since that work was published. TOKAN has a large number of other features that allow the user to perform different kinds of orthographic normalizations or control how the output is ordered and presented as it may fit different needs of different systems. Table 1.14 exemplifies a few of the TOKAN supported schemes with the same sentence.

Internally, TOKAN uses morphological generation (through ALMORGEANA) to recreate the word once different clitics are split off. This approach of back generation allows us to modify the morphological content in a word including, for instance, deleting/defaulting specific features of a word easily. We do this to guarantee the form of the generated word is normalized and consistent with other occurrences of that word. For example, simply splitting the pronominal clitic off a word with Ta-Marbuta (ة *p*) would keep the Ta-Marbuta in its word-internal form (regular letter Ta, ت *t*). With TOKAN, the Ta-Marbuta is generated as appropriate (e.g., جولته *jwlth* ‘his-visit’ is tokenized into جولته *jwlp+h* ‘visit +his’, not جولت *jwlt+h* (which is not a valid spelling).

1.4.2.6. AMIRA

AMIRA is a set of tools built as a successor to the ASVMT toolkit developed at Stanford University (Diab *et al.* 2004) and described in detail by Diab *et al.* (2007b). The toolkit includes a tokenizer, a part of speech tagger (POS) and a base phrase chunker (BPC), also known as a shallow syntactic parser. The technology of AMIRA is based on supervised learning with no explicit dependence on knowledge of deep morphology, hence, in contrast to the previous tools, it relies on surface data to learn generalizations. In general the tools use a unified framework, which casts each of the component problems as a classification problem. The underlying technology uses Support Vector Machines in a sequence-modeling framework using YAMCHA.²² The AMIRA tools are highly accurate, very fast, and robust. They allow for a limited number of variable user settings depending on the disambiguation granularity. The AMIRA tools have been widely used for different NLP applications due to their speed and high performance, such as MT, IR, Parsing, NER, and IE.

From the user perspective, the different components of the tool suite could be invoked sequentially, taking raw text in any encoding and producing clitic tokenized, POS tagged and/or base phrase chunked data. They could also be applied directly on some given text. For example POS tagging could be applied on raw text without explicitly invoking tokenization. We briefly review the three main components of the AMIRA suite.

1.4.2.7. AMIRA-TOK

A tokenization scheme is a convention chosen by the researcher, and there are many possibly tokenization schemes. For AMIRA-TOK we focus primarily on clitic tokenization. Clitics in Arabic are typically affixes that latch onto base words. They are morphemes that have a syntactic role and affect the orthography and phonology of words. AMIRA tools do not rely on morphological analysis or generation tools in any of its processes.²³ Hence, AMIRA-TOK learns clitic tokenization generalizations from the clitic segmentations present in the Penn Arabic Treebank (PATB) directly without relying on rules explicitly.

AMIRA Clitics: In addition to the standard PATB clitic tokenization, AMIRA-TOK can also segment off the nominal definite article *Al*. Accordingly, we have modified the training data to include segmentation of the *Al*.²⁴ The motivation behind segmenting off the *Al* is that it has a positive impact on higher order NLP applications especially those involving correspondence to other languages where the definite article is a stand alone word. Moreover, this segmentation reduces the sparseness associated with nominal forms.

In total, AMIRA-TOK segments off the following set of clitics: conjunction proclitics (prefixes) *w*, *f*, prepositional proclitics *k*, *l*, *b*, future marker proclitic *s*, verbal particle proclitic *l*, definite article proclitic *Al*, and pronominal enclitics (suf-

²² <http://chasen.org/~taku/software/yamcha/>

²³ For example, the definite article preceding sun letters in Arabic.

²⁴ A clear demarcation of the definite article *Al* is present in the fully vowelized version of the PATB.

fixes) indicating possessive/object pro-nouns. In general, nominals can have 0- 3 possible proclitics from the conjunction, prepositions and definite article set. Verbs can have 0- 2 proclitics from the set of conjunctions and the future clitic marker.²⁵

Since multiple proclitics are allowed in Arabic, they follow a specific order if present. The definite article ال *Al* is always closest to the base word, followed outwardly by the prepositions, then by the conjunctions. Likewise for the verbal proclitics, the future marker or verbal particle is the clitic affixed to the verb directly, followed outwardly by the conjunction. Accordingly, in the example *و بالحسنات و bAlHsnAt*, ‘and by the virtues’, the proclitic tokenization will be rendered as follows: *و w+ب b+ ال Al+ حسنات HsnAt* where *و w* is the conjunction and, *ب b* is the preposition ‘by’, and *ال Al* is the definite article ‘the’.

The modeling of clitic tokenization in AMIRA-TOK is exactly the same.

Character Chunking: For AMIRA-TOK, we apply a chunking scheme on the character level casting the problem as a chunk boundary identification and chunk classification problem. We use an IOB annotation scheme, every character in our data (including punctuation) is annotated as: inside a chunk (I), outside a chunk (O), or beginning of a chunk (B). For the I and B tags, we have five possible classes: Pre-fix 1, Prefix 2, Prefix 3, Word, Suffix. This leads to a total of 11 classes in the data: O, B-PRE1, I-PRE2, B-PRE2, I-PRE2, B-PRE3, I-PRE3, B-WORD, I-WORD, B-SUFF, I-SUFF.

Given raw data, the words are segmented into a series of chunks corresponding to the different classes above. Our goal is to produce text where all the word tokens (modulo the clitics) are bona fide words in a dictionary and increasing the level of regularity in the text as a means of normalization. Note that AMIRA-TOK does not perform lemmatization. For example, in our running example *و بالحسنات و bAlHsnAt*, ‘and by the virtues’, the clitic tokenization AMIRA-TOK reduces it maximally to *حسنا + ال + ب + و + w+ b+ Al+ HsnAt*, where *حسنا HsnAt* is kept in its plural form and not reduced to the singular lemma form *حسنة Hsnp*.

Morphotactic Restorations: AMIRA-TOK does not produce stemmed words, which are not valid Arabic words. Accordingly, we apply some heuristics to reverse the effect of morphotactics such as the loss of *l A* in the definite article *ال Al* when in the context of the proclitic preposition *ل l* ‘for’. For example, the input word *وللبلاد wllblAd*, ‘and for the countries’, is clitic tokenized as *w+l+l+blAd* in the system output from the classification. A post-hoc fix is applied to ensure the consistency of the output. The final output is rendered *بلاد+ال+ل+و w+l+Al+blAd*. Most of these morphotactic restorations can be deterministically applied.

However, some morphotactics are not deterministic such as those involving the nominal feminine marker (Ta-Marbuta), the Alef-Maqsura, and word final Hamzas. In AMIRA-TOK, we currently handle the former two cases: the Ta-Marbuta and the Alef-Maqsura.

²⁵ Prepositions and definite articles, by definition, do not affix onto verbs and similarly, future marker *س s* and verbal clitic *ل l* clitic do not prefix onto nominals and do not co-occur.

Ta-Marbuta Normalization: The Ta-Marbuta (ة *p*) is a word final nominal feminine marker that is realized as a ت *t* preceding a pronominal enclitic. Accordingly, a stem after token cliticization ending with a ت *t* could either be a verb (not a Ta-Marbuta) or a noun (possibly a Ta-Marbuta). In *w+b+Hsnt#hm*, ‘and by their virtue’, the word-final ت *t* in *Hsnt* is a ة *p* converted to ت *t* in the context of the enclitic *hm*: *w+b+Hsnt#hm*. As part of AMIRA-TOK, the ت *t* is converted to the corresponding ة *p*. This allows for more accurate POS tagging later as the base word *Hsnp* will correctly be identified as a noun rather than verb.

Alef-Maqsura Normalization: The Alef-Maqsura (ى *Y*) always occurs word finally. It is letter that graphically looks like a dotless Ya (ي *y*), yet phonologically sounds like an Alef (أ *A*). In the context of a pronominal enclitic, it changes to an orthographic أ *A*. For example a noun such as *ESAhm* ‘he-went-against them’, is initially segmented into *ESA#hm* yet the stem final أ *A* should be an Alef-Maqsura (ى *Y*): *ESY#hm*. In this case the resulting verb stem *ESY* would be confused with the noun *ESA* ‘stick’. In most cases the variation of ي *Y* and أ *A* is a lexical distinction relating to the underlying root of the word or the word pattern. Accordingly, AMIRA-TOK converts these word final أ *A* into ي *Y*.

Both Ta-Marbuta and Alef-Maqsura normalization are lexically determined; hence, we apply another layer of learning to the problem of classifying word final letters. A stem final ت *t* either remains a regular ت *t* or is converted to ة *p*, and the stem final أ *A* either remains an أ *A* or is converted to a ي *Y*. As a result, the end output from the AMIRA-TOK is valid surface form words with clitic tokens. AMIRA-TOK performs at a very high F-score measure of 99.2% evaluated using standard Conllev evaluation metrics.

User Interface: In the current version of AMIRA-TOK, the user is allowed ample flexibility in deciding the form of output they would like. The user can decide what level of clitic segmentation they desire, and whether tokenization should be indicated with spaces (changing the token count) or with a plus sign only (preserving token count). We have specified several schemes as illustrated in Table 1.15.

	wbAlHsnAt	wllblAd	Fbmkbtbthm	wsnqwlhA
Arabic	وبالحسنات	وللبلاد	فبمكتبتهم	وسنقولها
Gloss	and+by+the+	and+for+the	then+by+libraries	and+will+we_s
Translation	virtues	+ countries	+their	ay+it
	And by the	And for the	Then by their	And we will say
	virtues	countries	libraries	it
Scheme				
PATB	w+ b+ AlHsnAt	w+ l+ AlblAd	f+ b+ mktbt #hm	w+ s+ nqwl #hA
AMIRA-TOK	w+ b+ Al+ HsnAt	w+ l+ Al+ blAd	f+ b+ mktb_h #hm	w+ s+ nqwl #hA
Conjunction-only	w+ bAlHsnAt	w+ llblAd	f+ bmkbtbthm	w+ snqwlhA
Conjunction-only+Suffix	w+ bAlHsnAt	w+ llblAd	f+ bmkbtb_h #hm	w+ snqwl #hA
Preposition-only	w+b+ AlHsnAt	w+l+ AlblAd	f+b+ mktbtbthm	wsnqwlhA
Preposition-only+Suffix	w+b+ AlHsnAt	w+l+ AlblAd	f+b+ mktb_h #hm	wsnqwl #hA
Al-only	wb+Al+ HsnAt	wl+Al+ blAd	Fbmkbtbthm	wsnqwlhA
Al-only+Suffix	wb+Al+ HsnAt	wl+Al+ blAd	fbmkbtb_h #hm	wsnqwl #hA
future marker-only	wbAlHsnAt	wllblAd	Fbmkbtbthm	w+s+ nqwlhA
future marker+Suffix	wbAlHsnAt	wllblAd	fbmkbtb_h #hm	w+s+ nqwl #hA
All Prefix Only	wbAl+ HsnAt	wl+ blAd	fb+ mktbtbthm	ws+ nqwlhA
Suffix only	wbAlHsnAt	wllblAd	fbmkbtb_h #hm	wsnqwl #Ha
All Prefixes+Suffix	wbAl+ HsnAt	wl+ blAd	fb+ mktbtbthm	ws+ nqwl #hA

Table 1.15: Some of the different AMIRA-TOK clitic tokenization schemes in the **change** token count mode

1.4.2.8. AMIRA-POS

Part-of-speech tagging is the process by which every word in running text is assigned a POS tag from a predefined tag set. AMIRA-POS assumes the text is clitic tokenized.

AMIRA POS Tag Sets: The POS tag set (ERTS) used by AMIRA-POS has 72 tags; it is a subset of the full morphological set defined over tokenized text. ERTS is a superset of the reduced PATB tag set (RTS) that comprises 25 tags. In addition to the information contained in the RTS tags, ERTS encodes additional morphological features such as number, gender, and definiteness on nominals. For example, in RTS nouns are tagged as either NN or NNS, indicating only number. In ERTS, nouns tags represent definiteness and gender in addition to number; furthermore, we add the dual as a value to the number feature. A full description of ERTS is presented by Diab (2007a).

POS Tagging: We adopt an SVM-based classification approach using character n-grams as features in our sequence models. Interestingly, the accuracy of the ERTS tagger is 96.13% and the accuracy of the RTS tagger is 96.15%. This suggests that our choice of information to include in our ERTS tag set reflects a natural division in the syntactic space. In previous work, (Diab 2007 and 2007a), we showed that using ERTS improves the quality of downstream processing such as base phrase chunking.

User Interface: The user has the flexibility to in-put raw or tokenized text in a scheme that is consistent with one of the schemes defined by AMIRA-TOK. Consequent-

ly the user may request that the POS tags be assigned to the surface forms. Internally, in case of the raw input, AMIRA-POS runs AMIRA-TOK on the raw text and then performs POS tagging. The output can be presented as tokenized and POS tagged, or without tokenization where the POS tag is assigned to the surface words. In this latter case, the ERTS tag set is appended with the clitic POS tags to form more complex POS tags. The user can choose to either tag with ERTS or RTS.

1.4.2.9. AMIRA-BPC

Base phrase chunking is the process by which a sequence of adjacent words is grouped together to form syntactic phrases such as NPs and VPs. An English example of base phrases would be [I]_{NP} [would eat]_{VP} [red luscious apples]_{NP} [on Sundays]_{PP}. BPC is the first step towards shallow syntactic parsing. Many applications such as information extraction and semantic role labeling in English have been proven to benefit tremendously from BPC at a relatively low loss in performance when compared to the use of deep syntactic parsing.

Chunking: AMIRA-BPC produces the longest possible base phrases with not much internal recursion. The internal recursion is done as a deterministic post process. We have modified the BPC rules to be more appropriate for the Arabic language (Diab 2007). Nine types of chunked phrases are recognized using a phrase IOB tagging scheme (described earlier in Section 1.4.2.6); Inside (I) a phrase, Outside (O) a phrase, and Beginning (B) of a phrase. The nine chunk phrases identified for Arabic are ADJP, ADVP, CONJP, PP, PRT, NP, SBAR, INTJ, VP. Thus the task is a one of 19 classification tasks (since there are I and B tags for each chunk phrase type, and a single O tag). The training data is derived from the PATB using the CHUNKLINK software.²⁶ CHUNKLINK flattens the tree to a sequence of base (non-recursive) phrase chunks with their IOB labels. For example, a token occurring at the beginning of a noun phrase is labeled as B-NP. We have adapted the CHUNKLINK software to the Arabic language.

However we do acknowledge that an appropriate size of a chunk is most dependent on an end application.

The AMIRA-BPC component is very fast and extremely accurate. The system yields an F1 mea-sure of 96.33%. Different research groups have shown the utility of BPC for different applications such as Machine Translation and information ex-traction specifically in Named Entity Recognition (NER) (Benajiba *et al.* 2008; Benajiba *et al.* 2009).

User Interface: AMIRA-BPC can accept any level of preprocessing on input text. AMIRA-BPC internally uses ERTS POS tag set. However, the user may request the RTS as the POS tag set, we have an internal mapping process from ERTS to the RTS POS tag set. Also AMIRA-BPC can produce the BPC tags on any form of AMIRA-TOK consistent schemes in addition to raw input text.

²⁶ <http://ilk.uvt.nl/team/sabine/chunklink/README.html>

1.4.2.10. Comparison of CADIM Tools

In this section, we compare and contrast the different CADIM tools discussed so far in terms of their design, functionality and performance.

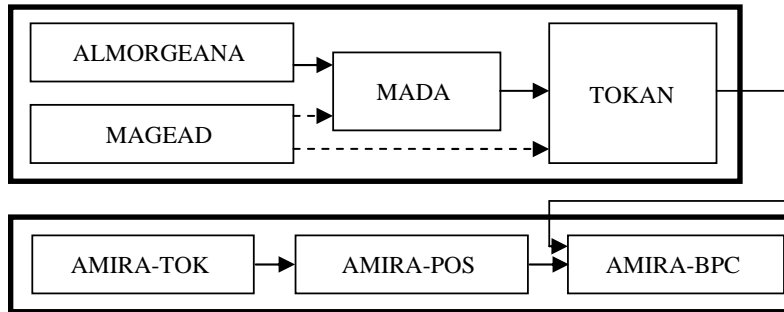


Figure 1.11. The MADA suite and the AMIRA suite.

Design: As for their design, it may help to contextualize the different tools in terms of their basic use in two suites: The MADA suite and the AMIRA suite (see Figure 1.11). Within the MADA suite, we have an explicit morphological analysis step handled by ALMORGEANA or MAGEAD. Currently, ALMORGEANA is fully integrated and we plan to integrate MAGEAD. MAGEAD uses a different approach to morphology than ALMORGEANA. The difference, however, should not be evident for MSA. MAGEAD’s advantage over ALMORGEANA is the ease of extensibility to new dialects. The second, in fact core, component in the MADA suite, is the MADA system, which disambiguates the analyses produced by the morphological analyzers. Finally, the TOKAN component makes use of the morphological generation power of ALMORGEANA and MAGEAD to tokenize the disambiguated analysis through regeneration.

In the AMIRA suite, the three components focus on tokenization (AMIRA-TOK), POS tagging (AMIRA-POS) and base-phrase chunking (AMIRA-BPC), in that order.

The AMIRA-BPC component is unique in that it has no parallel in the MADA suite. In fact the output of TOKAN can be designed to look like the output of AMIRA-TOK followed by AMIRA-POS, and then be used as input to AMIRA-BPC.

In terms of their design, AMIRA-TOK and AMIRA-POS are different from the MADA suite in that they take a two-step approach to POS tagging: tokenize then tag. In comparison, MADA has a different approach that breaks the problem into three steps (analyze, disambiguate, generate), which are orthogonal to AMIRA’s split. In the AMIRA suite, the three components focus on tokenization (AMIRA-TOK), POS tagging (AMIRA-POS) and base-phrase chunking (AMIRA-BPC), in that order.

Although there are three steps in MADA, the decision for tokenization and POS tagging is done together in one-fell-swoop. One way of distinguishing these tools is in terms of the depth of linguistic knowledge needed. AMIRA is shallow in that it focuses on form-based morphology (specifically cliticization) learned from annotated data; whereas MADA uses deeper lexically modeled functional morphology. Another difference between the current MADA suite and the AMIRA-TOK and AMIRA-POS suite is that the former may produce no analysis for a given word if it does not exist in the underlying

morphological tools while the AMIRA suite always produces a hypothesized tokenization and POS tag for every word in the text.

In terms of their training needs, the MADA suite expects the presence of both a morphological analyzer and training data for supervised learning, whereas the AMIRA suite only needs annotated training data. The training data could be created through any number of ways, including the use of morphological analyzers followed by human annotation as is done at LDC; but this is not a requirement for the AMIRA suite. These different yet similar requirements put similar limits on the kind of extensions that could be done in either approach. For example, going to an Arabic dialect would require the presence of some morphological analyzer/generator for the dialect for MADA, but not AMIRA. However, both need some amount of annotated data to train on.

Functionality: Base-phrase chunking is only handled in the AMIRA suite, but it is in fact a separate module that can be used independently with the MADA suite. The other four applications are handled at once in MADA as part of its common morphological disambiguation process. AMIRA does not handle lemmatization or diacritization. As for tokenization and POS tagging, since MADA goes deeper than AMIRA, a wider set of possible tokenization schemes and POS tags can be output by MADA. Although AMIRA is more limited by comparison, it does handle the most commonly used tokenizations and POS tags. Researchers interested in exploring a large number of different sets of tokenizations as features in their systems should consider MADA. Researchers only interested in limited comparisons or specific applications, whose tokenizations and POS tags are supported by AMIRA should consider AMIRA.

Performance: It is hard to compare the performance of AMIRA and MADA suites. Previous attempts by Habash and Rambow (2005) show that similar performance is possible on tasks that are shared: specific PATB tokenization and POS tags. AMIRA can be significantly faster than MADA; however, MADA needs to be run only once and a much larger number of tokenizations and POS tags (in addition to other outputs not supported by AMIRA) can be produced by running the fast TOKAN step.

Applications: In terms of functionality, we consider five applications: tokenization, diacritization, POS tagging, MADA, TOKAN, and the AMIRA suites have been and are being used by 50+ academic and commercial research institutes around the world, including University of Washington, University of Massachusetts, University of Maryland, University of Sussex, Dublin City University, Cambridge University, SRI, BBN, Fair Isaac Inc., SketchEngine LLC, MIT, RWTH Aachen, Polytechnic University of Catalunya (UPC), Copenhagen Business School, Polytechnic University of Valencia (UPV), and the National Research Center of Canada, among others. The tools have been cited in numerous publications and have been shown to improve performance in a variety of NLP applications such as MT, IR, and IE.

MADA+TOKAN for NLP applications: In the context of machine translation (MT) from Arabic to English, Habash and Sadat (2006) and Sadat and Habash (2006) explored the use of different preprocessing schemes and their combination. Their results have been

followed by different groups of researchers working on Arabic-English MT (Costa-jussà *et al.* 2006; Crego *et al.* 2006; Vilar *et al.* 2008). Diab *et al.* (2007a) explored the use of MADA-generated diacritizations for MT. Elming and Habash (2007) and Elming *et al.* (2008) improved automatic word alignment for Arabic-English MT using combinations of different tokenization schemes generated by MADA+TOKAN. See (Habash 2007) for more details on different representations of Arabic morphology for MT. Badr *et al.* (2008) used MADA in the context of English-to-Arabic MT. MADA has also been used to produce features for Named Entity Recognition (NER) (Farber *et al.* 2008; Benajiba *et al.* 2008).

AMIRA for NLP applications: The AMIRA suite has been successfully used by several groups in the context of text MT, specifically for alignment improvement and reordering within the context of statistical MT (Crego and Habash 2008), and also for identifying difficult source language text (Mohit and Hwa 2007). Moreover, the AMIRA suite was used in the context of speech MT (Stroppa and Way 2006). The AMIRA suite was explored for the purposes of cross language information retrieval in work by Larkey *et al.* (2007). AMIRA has been used to produce POS tag and BPC features for NER, which significantly boost the performance of the Arabic NER system (Benajiba *et al.* 2008; Farwell *et al.* 2007).

Case Study of a Complex ASR+MT system: In the following example we show how MADA+TOKAN can be incorporated into an MT project that is based loosely on the SRI GALE project Nightingale. Figure 1.12 shows the overall architecture of the project. The MT process needs to make use of data from both text sources and audio sources via automatic speech recognition (ASR). The ASR process processes audio files, but needs to build reliable language models from text sources first. The subsystems ASRLM and MTTEXT process raw Arabic script before passing important information to the ASR and MT components, respectively. The MTASR subsystem processes ASR output for use in MT. These three subsystems use specific instantiations of MADA+TOKAN with different input and output specifications.

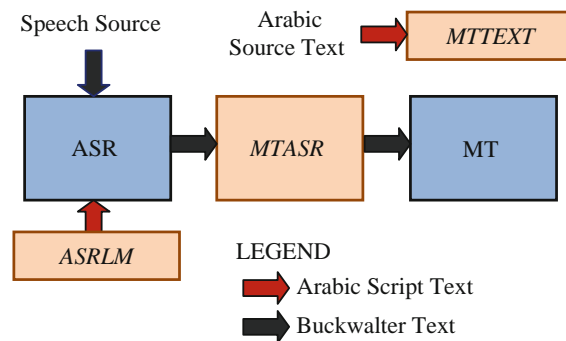


Figure 1.12. Example of an ASR/MT pipeline. MADA and TOKAN are used in the ASRLM, MTASR, and MTTEXT subsystems.

The ASRLM subsystem cleans the raw data and converts the UTF-8 encoding into Buckwalter transliteration. A separate utility is used to convert numeric digits to word expressions (Habash and Roth 2008), as is required for ASR. The subsystem then runs MADA and uses the toolkit’s stem orthographic normalization tool to remove spelling variations. The subsystem consequently runs TOKAN to produce an output suitable for ASR; here, TOKAN uses a basic readoff scheme that produces only the fully diacritized words without further tokenization. Finally, punctuation is removed. This provides the ASR system with nicely formatted, fully diacritized data, which is what the acoustic component of the ASR component produces.

The MTASR subsystem takes the output of ASR (which originally came from the audio files), cleans it, and runs MADA+TOKAN, using a tokenization scheme that splits off conjunctions and particles, (a tokenization scheme known as ‘D2’). Stem orthographic normalization is also used to collapse words with common spelling variants. The same numerical utility used in ASRLM is also used here to tag numerical expressions (which may be digital or expressed as words). The Buckwalter-transliterated data is converted back to UTF-8 prior to sending the data to the MT system.

The MTTEXT subsystem processes text for MT. It cleans the raw data and converts UTF-8 to Buckwalter transliteration. MADA+TOKAN (with stem orthographic normalization, number tagging and UTF-8 conversion) are used here to produce the same tokenization (D2) as the MTASR subsystem, making the output of MTTEXT and MTASR identical. The result is that the MT system can draw on similarly formatted ASR-derived and text-derived data for training and development.

1.4.3. Chinese Statistical Parsing

Authors: Mary Harper and Zhongqiang Huang

1.4.3.1. Introduction

This section describes several issues that are fundamental to achieving accurate Chinese parsing given available Chinese resources and the challenges of the Gale processing pipeline. For Gale, our parsing algorithm was expected to accurately parse various different materials, ranging from newswire text, which tends to be grammatically well formed, to n -best ASR outputs, many of which are poorly formed sentences. To address this challenge, we have re-implemented and enhanced the Berkeley parser to handle unknown Chinese words efficiently, parse difficult sentences robustly, and operate more efficiently. We also address issues related to training the parser for several different genres given a limited number of available training trees, the importance of matching word segmentation to the treebank segmentation standard to support accurate parsing, and the need for standardized tokenization for managing the types of things that will appear as input to the parser. Understanding and handling these issues is a prerequisite for achieving adequate parsing performance levels. We also investigate self training with automatically labeled in-domain data to enhance parsing performance given the limited number of trees in the Chinese treebanks.

There have been several attempts to develop high quality parsers for Chinese (Bikel and Chiang 2000; Levy and Manning 2003; Petrov and Klein 2007a), but the state-of-the-

art performance, around 83% F measure on Penn Chinese Treebank, achieved by the Berkeley parser (Petrov *et al.* 2006; Petrov and Klein 2007a) falls far short when compared to English.²⁷ As pointed out by Levy and Manning (2003), there are many linguistic differences between Chinese and English, as well as structural differences between their corresponding treebanks, and some of these make it a harder task to parse Chinese. Additionally, the fact that the available treebanked Chinese materials are far more limited than those for English also increases the difficulty of building high quality Chinese parsers. While there have been some investigations using treebanks to determine what makes Chinese hard to parse, here we attempt to learn about and control the factors that challenge Chinese parsers as applied to materials that are less controlled than a treebank.

Research that involves only parsing treebank test sets in some sense masks the challenges of providing high quality syntactic analyses for Chinese text and speech. First, Chinese documents are not segmented into words that conform to the word segmentation standard used by the Chinese treebank. Second, the Chinese treebanks contain punctuation, letters, and digits that are typically full-width; whereas, it is not uncommon for materials to contain mixtures of full- and half-width representations. It is in fact unlikely that the treebank would cover the combination of possible character types that may be observed in Chinese documents. Third, the treebank data is quite limited compared to the resources available for English parsing of newswire and speech genre, and so we must develop strategies to effectively parse materials in the text and speech genre covered by the GALE program given the resources at hand: Chinese Treebank 6 (CTB6) with parsed newswire and broadcast news sentences and Chinese Broadcast Conversation Treebank (BCTB) with parsed broadcast conversation sentences.

We will focus our parsing investigations in this section on the Chinese newswire and broadcast conversation genres because of the availability of tree-banks representative of these genres. Unfortunately, the broadcast news treebank does not fully capture the spoken aspects it should represent in that it contains symbolic expressions that can be verbalized in a variety of ways. Although we can map this treebank to a verbalized form, this mapping is unfortunately less variable than an exact transcription of the corresponding speech would be. To maximize parsing performance, we process the treebanks differently for the newswire and broadcast conversation genres and train the models to match genre conditions. We evaluate genre matched parsers on a test set drawn from the respective newswire and broadcast conversation treebank.

1.4.3.2. Experimental Data

We used the Penn Chinese Treebank 6.0 (CTB6) and the Chinese Broadcast Conversation Treebank (BCTB) in our parsing studies, as well as a set of unlabeled sentences to support the use of semi-supervised self-training (see Table 1.16).

CTB6 contains 28k parsed sentences, including news articles from Xinhua news agency (China-Mainland), Information Services Department of HKSAR (Hong Kong), and Sinorama magazine (Taiwan), as well as broadcast news from ACE evaluation (which we call CTB6BN). The news articles from the first three sources, with a total of

²⁷ In some cases, the parse results do not take into account the brackets of gold trees for sentences that do not produce a final parse in the score, and so the reported results are more optimistic than they should be.

19k sentences, constitute the former Penn Chinese Treebank 5 (CTB5) and are the primary source of labeled data used for newswire experiments. Since the CTB5 corpus was collected during different time periods from different sources with a diversity of topics, in order to obtain a representative split of train-test-development sets, we divide it into blocks of 10 files in sorted order and for each block use the first file for development, the second for test, and the remaining for training. Although CTB6BN consists of parses for broadcast news transcriptions, it exhibits many of the characteristics of newswire text (it contains many nonverbal expressions, e.g., numbers and symbols, and is fully punctuated). Because of this similarity, we evaluate using this corpus as additional labeled training data for training the newswire model in Section 1.4.3.7.

Genre	Train	Dev	Test	Unlabeled
NW	CTB5 14,925 (~405k)	CTB5 1,904 (~51k)	CTB5 1,975 (~52k)	210k (~6255k)
	CTB5+CTB6BN 24,416 (~679k)			
BC	BC 8,149 (~110k)	BC 1,297 (~14k)	BC 1,497 (~17k)	210k (~2128k)
	CTB5+CTB6BN 24,412 (~584k)			
	BC+CTB5+CTB6BN 32,561 (~695k)			

Table 1.16. Number of sentences and words (in parentheses) of the data splits used in our experiments.

BCTB contains 11k parsed sentences from three broadcast conversation sources (i.e., CCTV, CNN with Chinese translations, and Phoenix TV). In contrast to the CTB6BN data, the words in this treebank are fully verbal; however, most of the sentences in the treebank are punctuated. We manually selected the train-test-development split of the files to balance the number of sentences in each set. Since BCTB may be too small to train an accurate grammar for parsing speech, we also augment the training set with a verbalized version of CTB6, as discussed in Section 1.4.3.7.

In addition to the above two labeled treebanks, we also utilize a greater number of unlabeled sentences for investigating the use of semi-supervised self-training in Section 1.4.3.8. For the newswire experiments, 210k unlabeled sentences are selected from three newswire sources covering materials from China Mainland, Taiwan, and Hong Kong. For broadcast conversation experiments, 210k unlabeled sentences are selected from transcribed broadcast conversations.

We have developed a set of scripts to clean up the treebank trees used in our investigations. We remove comments and then delete empty nodes and non-terminal-yield unary rules, e.g., NP→VP, using *tsurgeon* (Levy and Andrew 2006). As non-terminal-yield unary rules are less likely to be posited by a statistical parser, it is common for parsers trained on the standard Chinese treebank to have substantially higher precision than recall. This gap between bracket recall and precision is alleviated without loss of parse accuracy by deleting the non-terminal-yield unary rules.

1.4.3.3. SParseval Scoring

The SParseval tool (Harper *et al.* 2005; Roark *et al.* 2006) was originally designed to support English speech-based bracket and head dependency scoring (recall, precision, and F-score) at the level of a demarcated chunk of speech such as a conversation side, while also supporting more traditional text-based scoring methods that require the input to the parser to match perfectly in words and sentence segments to the gold standard. The tool was developed to address the fact that output from an automatic speech recognition system is likely to contain word errors and the automatic segmentation of these words into sentences is likely to differ from those in the gold standard parses (Harper *et al.* 2005). To calculate the bracket scores in the face of word and sentence segmentation errors, the tool utilizes information from a word-level alignment between the yields of the system parses and reference parses in a transcript chunk (e.g., a conversation side or story).²⁸

We extended the functionality of this tool to support the scoring of Chinese parses when the word segmentation of the input to the parser differs from the gold standard word segmentation in order to investigate the effect of word segmentation algorithms on parse quality. We use bracket scores with word alignment as the metric for evaluating word segmentation impact. To score parses with different word segmentations using this tool, we align the words in a gold standard file with the words in a test file and score the brackets with punctuation removed.²⁹

1.4.3.4. Parsing Model

The Berkeley parser (Petrov *et al.* 2006; Petrov and Klein 2007a) is an efficient and effective parser that introduces latent annotations (Matsuzaki *et al.* 2005; Prescher 2005) to refine the syntactic categories and learns PCFG grammars based on these latent annotations. We have re-implemented and enhanced the Berkeley parser to handle unknown Chinese words efficiently, parse difficult sentences robustly, and operate more efficiently.³⁰ We will describe our enhancements in detail in the remainder of this section.

1.4.3.4.1. Unknown Word Handling

The Chinese word formation process can be quite complex (Packard 2000), and it differs substantially from the English process. Although Chinese morphology is generally considered less informative of the part-of-speech (POS) type of the entire word, its characters do reflect some information about a Chinese word. The last characters in a Chinese word are, in some cases, most informative of the POS type, while for others, it is the characters at the beginning or the middle of a word.

²⁸ The tool can also provide scores based on all of the head dependencies extracted from the system and reference trees, as well as a more focused set of open class dependencies, which involve open class content words.

²⁹ When evaluating test parses that have the same word segmentation as the gold standard parses, SParseval provides scores on a sentence-by-sentence basis, just like EVALB.

³⁰ The major motivation for re-implementation is to allow more generic and flexible state-tying operations that are important for some algorithms we are developing.

The Berkeley parser has some built-in ability to handle certain classes of unknown Chinese words such as digits and dates. For words outside of these classes, the parser ignores character-level information and relies only on the rare-word part-of-speech tag statistics. In our approach, which is similar to (Huang *et al.* 2007), we employ a rather simple, but effective method to estimate the word emission probability, $p(w|s)$, of an unknown word w given the latent state s of some syntactic category. We use the geometric average of the emission probability of the characters in the word, i.e., $p(c_k|s)$ with c_k being the k -th character of the word, to estimate the word emission probability:

$$P(w|s) = \sqrt[n]{\prod_{c_k \in w, P(c_k|s) \neq 0} P(c_k|s)}$$

where

$$n = |\{c_k \in w | P(c_k|s) \neq 0\}|.$$

characters unseen in the training data are ignored in the computation of the geometric average. In case there is any character in the word that was previously seen in the training data, but only with other latent annotations, then we back off by using the rare word statistics from the state s regardless of the word.

As we can see in Table 1.17, the character based unknown word handling method improves performance on both recall and precision when evaluated on CTB5.

Trebank	Unk	Recall	Precision	F
CTB5	default	82.12	82.88	82.50
	character	82.84	83.22	83.03

Table 1.17. The effect of the character-based unknown word handling method.

1.4.3.4.2. Robustness

Although the grammars trained by the Berkeley parser are compact, parsing can still be computationally expensive because of the many fine-grained latent states to consider in the computation of inside-outside probabilities in the chart. Fortunately, Petrov and Klein (Petrov and Klein 2007a) developed an efficient coarse-to-fine parsing strategy that starts from simpler grammars and prunes away unlikely chart items before parsing using the more complex grammars. In practice, the pruning thresholds are chosen to balance the parsing accuracy and speed.

Setting pruning thresholds to support robust but efficient parsing can be challenging over the range of parsing tasks we support in the GALE project; our parser is expected to parse various different materials, ranging from newswire text, which tends to be grammatically well formed, to n -best ASR outputs, many of which are poorly formed sentences. It is not uncommon for thresholds that work well for parsing treebank materials to fail to parse less well-formed sentences we are expected to process. To support more robust parsing, we incrementally lower pruning thresholds when the current thresholds fail to parse a sentence and then restore the default thresholds after finishing parsing that sen-

tence. With this strategy, we were able to more fully parse sentences in the GALE processing pipeline. We balance this with a parsing time threshold to address cases when sentences are too long to parse in a reasonable amount of time, returning an empty parse when the time threshold is exceeded.

In addition to variable pruning thresholds, we made several other minor changes to the parser to improve robustness. For example, we added smoothing for the word emission probabilities associated with a word tag pair w, t to eliminate zeros when a frequently observed word w does not co-occur with tag t in the training data.

1.4.3.4.3. Parallelization

The time needed to train a single-threaded Berkeley parser is acceptable for a corpus of moderate size. For example, it takes roughly one day to train a grammar on CTB5+CTB6BN. However, the training speed becomes a bigger issue when applying self-training strategies. For example, in the self-training experiments described in Section 1.4.3.8, the automatically labeled training data is an order of magnitude larger than the treebank training set, and it takes several weeks to finish training a more complex grammar. In our re-implementation of the parser, we parallelized the Expectation-Maximization step, the most computationally intensive component of the training code to take advantage of the computing power of multi-core/multi-CPU machines. The resulting training code is about 7 times faster with 10 threads on a 16-core machine than the single threaded version. We parallelized the parsing code as well to improve the decoding speed. The MapReduce framework (Dean and Ghemawat 2005) is also naturally applicable to both grammar training and sentence parsing, and this framework will be explored in future work.

1.4.3.5. Word Segmentation and Parsing

Written Chinese text consists of sequences of characters with no delimiters between words, and yet for word-based NLP applications (tagging, parsing, MT), word segmentation is a prerequisite. The output of word segmentation algorithms may vary depending on their different definitions of words and system engineering requirements. While word segmentation, in and of itself, is worthy of study, as shown in the efforts in the SIGHAN Chinese Word Segmentation Bakeoffs (Sproat and Emerson 2003), in this section we focus on the effect of word segmentation on parse accuracy. Discrepancies in word segmentation compared to the gold standard are likely to be the source of significant error when comparing the parse of those words to the gold standard parse of the gold standard word segmentation. Consistency between the words in the gold standard trees and the input to the parser is likely to be an important factor for achieving greater parse accuracies.

Here we will use the Penn Chinese Treebank (CTB) Segmentation Standard (Xia 2000) because that is the resource we use to train our Chinese parser (Xue *et al.* 2002; Xue *et al.* 2005). In the rest of this section, we will first describe the word segmentation algorithms we use and then evaluate the effect of those algorithms on parse accuracy. As described in Section 1.4.3.3., the Sparseval tool with alignment will be used to score

parses, because there can be mismatches between the yields of the test parses and gold standard trees.

1.4.3.5.1. Word Segmentation Algorithms

Parse accuracy can be substantially affected by the word segmentation algorithm and how it is trained. A word segmentation algorithm that is trained on treebank data is likely to produce a word segmentation that is more consistent with the treebank, and so is likely to result in parses with greater parse accuracy. We evaluated three different segmenters.

The first is Fair Issac's extension of the LDC segmenter (LNUplus) with an expanded dictionary (it adds words from the NYU name dictionary and the lexicon used by the University of Washington word segmenter), and instead of using a greedy left-to-right, longest dictionary entry match at each point, it uses dynamic programming to find a segmentation that maximizes mean token length. This is strictly a dictionary-driven segmenter, and the dictionaries it uses were not optimized to conform with CTB6.

The University of Washington word segmenter (UW) (Hwang *et al.* 2006) uses a very large segmentation dictionary that includes frequent words extracted from GALE acoustic data transcripts, Chinese Gigaword2, and the Chinese treebank 6.0. It then uses the longest-first-match algorithm to segment the training data comprised of GALE acoustic data transcripts and Chinese Gigaword, and then trains an n -gram LM on these sources and the Chinese treebank 6.0 data duplicated 5 times to weight the source more heavily. The language model uses a vocabulary of 70K words, with the remaining words mapped to a garbage word. This n -gram LM is then used to provide the maximum likelihood segmentation for a sentence. This word segmenter uses information provided by CTB6, but it is not optimized for the treebank.

The Stanford Chinese Word Segmenter (Tseng *et al.* 2005; Chang *et al.* 2008) uses conditional random fields with character identity, morphological, and character reduplication features extracted from the training data. The current segmenter also uses external lexicon features (Chang *et al.* 2008) to segment more consistently. We trained several versions of the Stanford segmenter:

- **Stanford (all)** is trained using the entire Chinese Penn Treebank. This represents an upper bound on the parsing accuracy for the parser given this word segmentation approach (since the training set contains the test sentences).
- **Stanford (parser)** is trained using the same training data split of the Chinese Penn Treebank as the parser, i.e., the CTB5+CTB6BN training set. The same dev set in Table 1.16 is used for development.
- **Stanford (parser+LDC)** is trained under the same conditions as Stanford (parser) except that the training data is augmented with an additional file that was segmented by LDC and contains 16,448 sentences.
- **Stanford (parser+LDC+self-labeled)** is trained under the same conditions as Stanford (parser+LDC) except that the training data is augmented with Gale data files that were segmented using Stanford (parser). Hence, this segmenter was trained on 175,852 sentences in total (24,416 from CTB5+CTB6BN training, 16,448 LDC hand-segmented, and 134,988 automatically word segmented).

- **Stanford (parser+LDC+UW-labeled)** is trained under the same conditions as Stanford (parser+LDC) except that the training data is augmented with the same Gale data files as in Stanford (parser+LDC self-labeled) segmented using the UW segmenter.

1.4.3.5.2. Results and Discussion

In Table 1.18, we compare the word segmentation and parsing performance of each of the segmentation algorithms and training conditions described in Section 1.4.3.5.1. The parser is our re-implementation of the Berkeley parser described in Section 1.4.3.4. The parser was trained on the CTB5+CTB6BN training set (i.e., treebank data only), and this parser was used to parse the CTB5 test set re-segmented by each of the word segmentation models. The resulting parses were then compared with the gold standard using aligned bracket scoring from the Sparseval tool as described in Section 1.4.3.3.

Models	Word Segmentation			Parser		
	Recall	Precision	F	Recall	Precision	F
Treebank Segmentation	100	100	100	83.58	84.00	83.79
LNUplus	81.70	82.50	82.10	67.46	68.29	67.87
UW	86.60	91.60	89.00	71.15	76.71	73.83
Stanford (all)	99.40	99.40	99.40	83.12	83.45	83.28
Stanford (parser)	95.70	96.60	96.20	80.49	81.65	81.07
Stanford (parser+LDC)	96.10	96.90	96.50	80.59	81.75	81.16
Stanford (parser+LDC+self-labeled)	97.50	97.60	97.50	81.84	82.27	82.05
Stanford (parser+LDC+UW-labeled)	89.00	90.50	89.80	74.13	75.98	75.04

Table 1.18: Results on the Chinese Treebank 6.0 test set for different models and training configurations.

It is not surprising that the best parsing performance is obtained using the treebank’s gold standard word segmentation. Parses of the sentences processed by the LNUplus and UW segmenters had significantly lower parse scores than was obtained given the gold standard word segmentation.³¹ Since neither segmenter was tuned to the treebank’s word segmentation standard, the errors in word segmentation harm the parse accuracies greatly. The UW segmenter’s use of the treebank data improves its word segmentation accuracy relative to the LNUplus segmenter; however, the use of training data that is inconsistent with the treebank’s standard plays a role in the 10% drop in parse accuracy.

To get at the upper bound parsing performance of the Stanford segmenter, we evaluated parsing accuracy when it was trained on the entire treebank. Although parse accuracy in this case is slightly lower than that obtained with gold standard word segmentation, it is clear that match to the gold standard words is an important factor for obtaining accurate Chinese parses. The Stanford segmenter was able to produce fairly accurate word segmentations when trained on the same treebank data as the parser and achieved parse F-scores within 3% of those obtained using gold standard word segmentations. It is notable that retraining the Stanford segmenter using additional LDC-segmented data and self-labeled data improves both word segmentation and parsing scores. When adding au-

³¹We used Bikel’s randomized parsing evaluation comparator to determine the significance ($p < 0.05$) of difference between two parsers’ output.

tomatically labeled data to the training set, it is important that it is consistent with the treebank word segmentation standard, as can be observed by the decline in performance obtained by adding data segmented by the UW segmenter. The use of consistently self-labeled data to re-train the segmenter improves parse performance by 1% F-measure over using the treebank training data alone. If parsing accuracy is important for downstream applications, then using a word segmenter that is tuned to the treebank standard is vital for achieving performance levels that are within 1-2% of those obtained with perfect word segmentation.

1.4.3.6. Text Tokenization

The UW Decatur (Zhang and Kahn 2008) text normalization process was developed to standardize the text pipeline stream for the Nightingale team. The value of this normalization for parsing lies with the fact that a parser that is trained to match its input conditions will perform better than a parser that must process highly varying data that is harder to model. For example, if we train the parser on the treebank, which largely contains full-width punctuation, but the input to the parser contains only half-width punctuation, then the parsing performance would be reduced due to the fact that the treebank contains only a few half-width punctuation tokens.³² While it is a simple matter to create a second set of training trees with all punctuation converted to half-width and mix it with the original treebank training, it may not cover all the ways in which full and half-width punctuation occur in real-world data. However, it is a simple matter to apply the Decatur normalization to our treebank by converting all full-width letters, digits, and punctuation marks to their half-width equivalents and train on the normalized trees. The Decatur tokenizer would then be applied to all input coming through the Nightingale pipeline to ensure that we have a good match between training and test conditions.

To ensure that the Decatur tokenization does not harm parsing performance significantly, we compare a parser trained on the original treebank CTB5 training set and tested on the original CTB5 test set with one trained based on Decatur normalized training and tested on the Decatur normalized CTB5 test set. As can be seen in Table 1.19, the parsing performance is slightly degraded by the normalization process, largely due to the fact that some punctuation distinctions are collapsed. However, because Decatur normalization will be carried out on all input to the parser, it eliminates a lot of the variability in the input that it would be hard to anticipate to adequately train the parser for the Gale pipeline.

CTB5	Recall	Precision	F
Original	82.84	83.22	83.03
Decatur	82.66	83.14	82.90

Table 1.19: Performance of the parser on CTB5 before and after Decatur normalization.

³²If we accidentally parse sentences using a model trained with a mismatching representation of punctuation, say the sentence contains full-width punctuation but the parser was trained with parses containing half-width punctuation, then the parse accuracy would be quite low, dropping from 82.90 to 72.16 F score. This reduction in parse accuracy results because the punctuation marks are not recognized as such.

1.4.3.7. Genre Mapping

Newswire stories contain a wide variety of textual phenomena, including words, symbolic expressions, and punctuation, using both full and half-width representations. When parsing newswire text, it is beneficial to normalize the input to the parser and map the training trees to the same normalized style to match the conditions in which the parser will be used. Additionally, it is important to fully utilize existing treebank resources to train the parser, given the limited amount of training data. Hence, we investigate the effect of combining trees from the broadcast news genre with the newswire training trees when training a newswire model. Table 1.20 shows the benefit of adding text normalized broadcast news data for training a better grammar for newswire. As we can see, the grammar trained on the combination of CTB5 and CTB6BN has a significantly better performance when evaluated on the CTB5 test set, due to the increase in domain matched training data. In an attempt to further boost performance, we will investigate the effect of combining automatically parsed newswire genre-matched trees with treebank training trees to further improve the newswire model in section 1.4.3.8.

Training Data	Recall	Precision	F
CTB5	82.66	83.14	82.90
CTB5+CTB6BN	83.36	83.95	83.65

Table 1.20: Results on the Chinese CTB5 test set for grammars trained on different configurations.

When parsing automatically transcribed broadcast conversation speech, the parser will need to process speech transcripts that are segmented into SUs using automatic sentence boundary detection; hence, words will be fully verbal (i.e., no symbolic expressions) and there will be no punctuation. While there is a small treebank that matches this genre, access to a greater diversity of speech-based trees is important for training as accurate a model as possible.

Because the annotated BC treebank is quite limited in size, a grammar trained only on BC treebank alone is quite poor as can be observed in Table 1.21. It is a natural idea to use CTB6 to enhance the training corpora for BC grammars. However, the textual nature of the CTB5 sub-corpus presents a challenge. One obvious difference is that the transcripts of the BC treebank are verbalized, i.e., all numbers, dates, and other phenomena are orthographically transcribed while CTB5 contains many symbolic forms. Although CTB6BN should be orthographically transcribed as speech, it contains digits and other symbolic expressions that make it inconsistent with the BC treebank. To address the mismatch between the CTB6 and BC treebanks in order to use CTB6 as additional training material for BC grammars, we have developed scripts to speech normalize (i.e., verbalize) the CTB6 treebank. The normalization procedure considers the part-of-speech tag of a word, as well as its context to determine how to verbalize the word. The major normalization operations include:

- Verbalize all sequences of digits based on their context. If the context suggests that a sequence of digits represents a number, this number is verbalized according to how it is pronounced, e.g., 12.4% is verbalized to 百分之十二点四(twelve

point four percent). Otherwise each digit is verbalized as it is pronounced individually, e.g., 03012 is verbalized to 零三零一二(zero three zero one two).

- Special care is taken to verbalize digits in temporal nouns. For example, 1998 is verbalized to 一九九八(year nineteen ninety eight), 5.4 to 五四(The May 4th Movement), 10:12 to 十分十二秒(ten minutes twelve seconds), etc.
- Split foreign names with first and last names delimited by a dot or hyphen into two names. Web and email addresses are also split into separate tokens with ‘.’ and ‘@’ verbalized.
- Remove all punctuation.

Table 1.21 shows the benefit of adding the speech normalized CTB6 for training a better grammar for conversational speech. As we can see, although the grammar trained on speech normalized CTB6 has a much lower performance when evaluated on the BC test set, its combination with the small amount of in-domain BC training data boosts the performance significantly, outperforming the grammar trained on BC training data alone.³³ We will also investigate the use of additional automatically parsed BC genre-matched sentences in the self-training manner to further improve parsing performance on BC data in section 1.4.3.8.

Training Data	Recall	Precision	F
BC	76.89	76.48	76.69
CTB5+CTB6BN	72.85	71.66	72.25
BC+CTB5+CTB6BN	78.95	78.96	78.96

Table 1.21: Results on the Chinese BC test set for grammars trained on different configurations.

1.4.3.8. Parser Self-training

To support self training in a semi-supervised setting, manually labeled training data is used to train an initial parser, which is then used to parse additional unlabeled data to combine with the manually labeled data to retrain a new parser. Early investigations using self-training for parser training were fairly unsuccessful at improving in-domain parsing. Charniak (1997) reported that self-training does not improve the performance of a context-free grammar trained on the WSJ training set. Steedman *et al.* (2003) reported some degradation using a lexicalized tree adjoining grammar parser (Sarkar 2001) and a minor gain using Collins lexicalized PCFG parser (Collins 1999); however, this gain was obtained using a poor parser trained on few sentences and was expected to level out quickly on a larger training set. One might conclude from these investigations that either the self-labeled data does not provide useful information or the training algorithm used for the parsers does not learn useful information from the self-labeled data.

Recently, McClosky *et al.* (2006) used semi-supervised training of a re-ranking parser (Charniak and Johnson 2005) and obtained exciting positive results on WSJ over a strong baseline. Their state-of-the-art re-ranking parser consists of two components, a

³³ If we had not normalized the treebank data for speech, the parsing accuracy would have been far worse due to a mismatch between the training and testing conditions.

lexicalized probabilistic 50-best parser and a discriminative reranker, which re-orders and selects the best of the 50-best parses returned by the first generative parser by utilizing a rich set of features that could not be feasibly used by the first parser. They used this two-stage parser to parse millions of unlabeled news article sentences and retrained the first parser using the combination of the original labeled data and the reranker-labeled data. Although they did not fully explain why this two stage method works, they suggest that the parse trees selected by the superior reranker provides guidance for retraining the first parser, which in turn produces better 50-best parses for the reranker. It should be noted that they also reported that no improvement was obtained from self-training their generative parser.

Genre	Training Data	Recall	Precision	F
NW	CTB6	83.36	83.95	83.65
	+unlabeled	84.28	85.28	84.78
BC	BCTB+CTB6	78.95	78.96	78.96
	+unlabeled	79.54	79.44	79.49

Table 1.22: The performance of self-training on both NW and BC.

We have investigated the self-training capability of our parser on both newswire and broadcast conversation by utilizing a moderately large amount of unlabeled in-domain data. As described in Section 1.4.3.2, 210k unlabeled sentences are used for both the newswire and broadcast conversation genres. The results are presented in Table 1.22. For the NW task, self-training gains one point on F measure over the grammar trained using the treebank CTB6 alone. This improvement is even greater than the benefit of adding CTB6BN to CTB5. The improvement on the BC task is smaller but consistent across different random runs. Given that the number of words for self-training BC is much smaller than for the NW task, use of additional genre-matched data is an important next step. All the improvements are statistically significant. We also tried self-training using a Chinese port of Charniak’s generative parser on our data set but obtained no significant improvement.

We observed that many of the rule parameters of the grammar trained on CTB6 alone have zero probability. On one hand, this is what we want because the grammar should learn about impossible rule expansions. On the other hand, this might also be a sign of over-fitting. In contrast, the grammar obtained using self-training contains far more non-zero rules than the grammar trained on CTB6 alone. This suggests that one of the benefits of using automatically labeled data is smoothing. The nature of the EM algorithm aims to adjust the model parameters to increase the likelihood of the training data. The greater the number of free parameters, the more power EM has to learn from and fit the training data. Resulting grammars may not generalize well when the training data is too small in size. We believe that the addition of automatically labeled data helps to prevent the EM algorithm from over-fitting the correctly labeled training data while learning with more latent states. This hypothesis will be investigated further in future work.

1.4.3.9. Conclusions

In this section, we have explored issues such as unknown word handling and word segmentation in parsing Chinese, and conducted experiments that highlight the fact that greater newswire parse accuracy can be achieved by training on the combination of newswire and broadcast news parses, than by training on newswire parses alone. Additionally, we have found that self-training with a large amount of unlabeled data further improves parsing performance. We conjecture based on our analyses that the EM training algorithm is able to exploit the information available in both gold and automatically labeled data with more complex grammars while being less affected by over-fitting the treebank. Self-training should also benefit other discriminatively trained parsers with latent annotations (Petrov and Klein 2007; Petrov and Klein 2008), although training would be much slower compared to using generative models, as in our case.

In future work, we will evaluate the impact of using larger quantities of self-labeled data. As the amount of data increases, it will also be important to investigate the impact of weighting the self-labeled data. It is quite possible that the errors in the automatically labeled data could limit the accuracy of the self-trained model, especially when there is a much greater quantity of automatically labeled data than the gold standard training data, so by weighting the posterior probabilities computed for the gold and automatically labeled data the information provided by the gold trees will not be swamped by the errors in the automatically labeled trees. Another approach would be to introduce the automatically labeled data at different stages of training rather than at the outset, because it is possible that a later introduction of the automatically labeled data, after well-founded annotations are learned from the treebank, would result in more effective learning. We will also investigate methods for automatic data selection of the automatically labeled sentences in order to choose those that would be most helpful for self-training.

1.4.4. Evaluating the Impact of Word Segmentation on Chinese Parsing

Author: Nianwen Xue

1.4.4.1. Introduction

Because Chinese does not have conventional word delimiters like white spaces, it is generally assumed that a prerequisite in Chinese language processing is to perform automatic word segmentation. Despite the enormous literature on Chinese word segmentation and syntactic parsing, to our knowledge, there has not been a thorough evaluation on the impact of word segmentation on Chinese parsing. As a result, there are often conflicting views on how much of a problem Chinese word segmentation is for later processing stages.

Word segmentation (or tokenization) has long been recognized as the first step in Chinese language processing and it is a problem that has generated a lot of research interest in the NLP community in the last few decades (Chen and Liu 1992; Sproat *et al.* 1996; Xue 2003; Gao *et al.* 2005; Low *et al.* 2005; Zhao and Kit 2008; Huang *et al.*

2008). There has been a steady improvement in word segmentation accuracy as the research shifted from early dictionary lookup approaches to statistical approaches (Sproat and Shih 1990; Sproat *et al.* 1996) and machine learning-based character-tagging approaches (Xue 2003; Gao *et al.* 2005; Zhao *et al.* 2006; Zhao and Kit 2008). It is still a very much active area of research in the Chinese language processing community, and the primary forum has been the successive SIGHAN Chinese word segmentation bakeoffs (Sproat and Emerson 2003; Emerson 2005; Levow 2006; Jin and Chen 2008).³⁴ Early successes came with the Maximum Match (Maxmatch) algorithm, a greedy search algorithm based on dictionary lookup. Given a text string, the algorithm looks for the longest word in the dictionary that matches the text from the current position. If there is a match, the current position is advanced past each character in that word. If there is no match, the algorithm starts again at the next character. There are some inherent problems with the Maxmatch algorithm. The performance of the Maxmatch algorithm is tied closely to the completeness of the dictionary. Since a complete dictionary is impossible to come by, given that new words such as names constantly find their way into the language, Chinese word segmentation is insolvable by the Maxmatch algorithm alone. In addition to new words, some character sequences are inherently ambiguous and the Maxmatch algorithm would consistently get it wrong in contexts where shorter words are the correct segmentation. For example, there are two perfectly legitimate segmentations for the character sequence “个-人”, depending on the context, e.g., “一/one 个/CL 人/person” vs. “个人/individual 所得税/income tax”. The Maxmatch algorithm would consistently get the latter but miss the former. To address the weaknesses of the Maxmatch algorithm, statistical approaches such as mutual information were proposed to capture the strength of the internal binding between the two characters within a word (Sproat and Shih 1990). However, mutual information based approaches have their own limitations. Mutual information for words longer than two characters is hard to define, and it is not straightforward to incorporate a dictionary in this framework, thus losing the benefit of a dictionary, in spite of the fact that no dictionary is complete. More recently, Chinese word segmentation has been formulated as a character tagging problem where each character in the text is labeled with a tag indicating its position within a word (Xue 2003). Combined with supervised machine learning methods this approach has consistently outperformed other approaches in the SIGHAN Bakeoffs (Xue and Shen 2003; Low *et al.* 2005; Zhao and Kit 2008) and produces state-of-the-art results. Top performing systems have consistently achieved accuracy that are in the high nineties (above 95%) using a wide range of benchmarks.

As large-scale hand-parsed Chinese corpora such as the Chinese Treebank (Xue *et al.* 2005) became available, there has been considerable work done on Chinese parsing as well (Levy and Manning 2003; Bikel 2004). However, Chinese parsers, at least by North America based researchers, are mostly extensions of English parsers. The same basic assumptions and methodologies are carried over from English to Chinese. With rare exceptions such as Luo (2003), most Chinese parsers assume word-segmented sentences as input and parsing accuracy is reported assuming gold standard word segmentation. In real-world applications, however, Chinese text has to be automatically segmented and there is inevitable degradation in parser performance caused by segmentation errors. As

³⁴ SIGHAN is a special interest group under ACL focusing on Chinese language processing. Website: www.sighan.org

far as we know, no formal evaluation on the effect of automatic word segmentation on parsing accuracy has been performed, although in informal settings in the GALE community there has been talk that parsing accuracy drops substantially when automatic word segmentation is used as input. The goal of this section is to elucidate this issue by performing a formal analysis and see how much impact word segmentation has on syntactic parsing.

1.4.4.2. Methodology

To perform such an analysis we obviously need i) training and test data, ii) a Chinese word segmenter, iii) a Chinese parser, and iv) an evaluation metric. For training and test data we use the Chinese Treebank 6.0 (CTB6.0).³⁵ CTB 6.0 has three main sources: Xinhua newswire, Sinorama magazine articles, and broadcast news.³⁶ The LDC release of CTB 6.0 has a recommended training set, development set, and test set and the data split can be found in CTB 6.0 documentation. The test set can be further divided into three subsets: the Xinhua newswire, the Sinorama and the broadcast news test sets. The training set has 639,697 words, consisting of roughly equal amount of Xinhua newswire, Sinorama and broadcast news data. The test sets also have roughly the same size: the Xinhua test set has 24,801 words, the Sinorama test set 28,879 words and the broadcast news test set 27,744 words. Both our word segmenter and parser are trained on the training set as a whole and tested separately on each of the three test sets. The Chinese parser we use is the Bikel parser (Bikel 2004) retrained on this training set without any further optimization, so the parsing results we present in this section may not be the best possible performance it can achieve. However, we believe that the performance is good enough so that whatever conclusion we draw from our experiments here will not be affected by the accuracy of the parser. The word segmenter is a Maxent-based Chinese word segmenter we developed in-house. The test data is first segmented by the Maxent word segmenter, and then parsed by the Bikel parser, and evaluated with the SParseval metric (Roark *et al.* 2006) and the CParseval metric we developed proposed. It is reasonable to assume that word segmentation accuracy will have a great impact on parsing accuracy as the latter takes the former as input, so it is important to use a state-of-the-art word segmenter so as to evaluate the impact of word segmentation on Chinese parsing given the current state of the art. It is also important to look at the evaluation metric very closely and examine the assumptions based on which the evaluation metrics are formulated. In the next two sections, we will look at these two aspects in greater detail.

1.4.4.3. Word Segmentation

The word segmenter we used for our experiment is an improved version of the word segmenter described by Xue (2003). The word segmenter formulates Chinese word segmentation as a character-tagging problem. The idea is to label each character in a sentence with its position in a word, and then reconstitutes the words in a sentence based on the character position labels. There are four possible positions for a character: in the be-

³⁵ Available from the Linguistic Data Consortium. Website: www ldc.upenn.edu

³⁶ A small amount of Hong Kong news has been folded into Xinhua newswire section in CTB 6.0.

ginning (LL), in the middle (MM), in the end (RR), or a word by itself (LR). Based on this formulation, the word sequence in (1a) would be labeled as (1b). The training data for the segmenter is derived from the training set in CTB 6.0 and a Maxent classifier is trained to tag un-seen data. The labeled data is then converted back to word sequences.

- a. 上海计到本世纪末实现人均国内生产总值五千美元
- b. 上/LL 海/RR 计/LL 划/RR 到/LR 本/LR
世/LL 纪/RR 末/LR 实/LL 现/RR 人/LL
均/RR 国/LL 内/RR 生/LL 产/RR 总/LL
值/RR 五/LL 千/RR 美/LL 元/RR
- c. Shanghai plans to reach the goal of 5,000 dollars in per capita GDP by the end of the century.

The features used in the (Xue 2003) are fairly simple and they are character unigrams and bigrams extracted from a five character window, consisting of the current character (C_0), the previous two characters (C_{-1} and C_{-2}), and the next two characters (C_1 , C_2), as well as, the tags of the previous two characters (T_{-1} , T_{-2}). The full list of features is described below:

- a. Character unigrams: current character (C_0), the previous (next) two characters (C_{-1} , C_{-2} , C_0 , C_1 , C_2)
- b. Character bigrams: the conjunction of the previous (next) character and the current character ($C_{-1}C_0$, C_0C_1), the previous two characters ($C_{-1}C_{-2}$) and the next two characters (C_1C_2), the previous and the next character ($C_{-1}C_1$)
- c. Tags: the tags of the previous two characters (T_{-1} and T_{-2})

This simple tagset is effective in predicting shorter words, but often makes mistakes with longer words such as numbers (e.g. 一百五十六点七八/1256.78), time expressions (e.g. 二零零九年/2009), idioms (e.g. 一目了然/clear at first sight) and names (e.g. 阿尔及利亚/Algeria), which often consist of four or more characters. To address these shortcomings, the improved segmenter used in our experiments also uses regular expression-based features, features extracted from a name list and an idiom list. These features are:

- a. The position of the current character (C_0) in a number, where the values for the position are “ln” (left of a number), “rn” (right of a number) or “mn” (middle of a number). For example, if the current position is 点, then its position is “mn” in 一百五十六点七八
- b. The position of the current character in a time expression, where the values for the position are “lt” (left of a time expression), “rt” (right of a time expression), or “mt” (middle of a time expression). For example, the position of 二 is “lt” in 二零零九年 if it is the current character.

- c. The position of the current character in an idiom, where the values of the position are “li” (left of an idiom), “ri” (the right of an idiom), or “mi”, (the middle of an idiom). For example, the position of 然 is “ri” in the idiom 一目了然
- d. Whether the current character starts or ends a name. For example, if the character sequence C_0C_1 matches the first two characters of a name, the feature “StartName” is extracted. If the current and the previous character match the last two characters of a name, then the feature “EndName” is added. For example, if C_0 =阿 and C_1 =尔, then the feature “Start-Name” is extracted given 阿尔及利亚 is in the name list.

Adding these features to the classifier cleaned up the errors with numbers, time expressions, names and idioms and improved the overall word segmentation accuracy significantly. Table 1.23 presents the word segmentation accuracy for each of the three test sets described in Section 1.4.4.2 when trained on the entire training set. For reasons that are not entirely clear, the accuracy for the Sinorama data is significantly lower than the other two sources, the Xinhua newswire and the broadcast news. The accuracy for the broadcast news is slightly lower, as expected because it is transcribed from speech data and is slightly noisier. Overall, the accuracy is comparable to that of the best systems in the SIGHAN bakeoffs. This allows us to measure the impact of word segmentation on syntactic parsing given the current state of the art.

genre	precision	recall	f-score
bn	95.36	95.51	95.44
xinhua	96.80	96.25	96.52
Sinorama	93.18	92.46	92.81

Table 1.23: Word Segmentation Accuracy.

1.4.4.4. Evaluation Metric: SParseval vs. CParseval

In over a decade, most of results on parsing are re-reported with the Evalb software using the Parseval metric, which measures the precision (the number of matched constituents between the treebank parse and the parser output to the total number of constituents in the parser output) and recall (the number of matched constituents to the total number of constituents in the treebank) and the F1 measure, which is a harmonic mean between precision and recall.

A key assumption for the Parseval metric is that there is a complete match between word tokens in a treebank parse and those in the parser output for a given sentence, an assumption that does not hold when parsing Chinese with automatic word segmentation as input.³⁷ One can imagine a trick that uses the characters instead of words as terminals in the parse tree. However, such a metric is essentially a combined metric of word segmentation and parsing, and given the reasonable assumption that word segmentation is

³⁷ <http://nlp.cs.nyu.edu/evalb>

substantially easier than syntactic parsing because the latter builds larger structures, the resulting accuracy is inflated and is incomparable with parsing accuracy reported for other languages using the Parseval metric.

The SParseval metric (Roark *et al.* 2006), which is developed to evaluate parsing accuracy on transcribed speech data and does not assume total match between word tokens in the gold standard parse and the parser output, seems like a natural alternative. So SParseval is the first evaluation metric we tried. When examining the evaluation results, we noticed that the metric penalizes word segmentation errors severely, even though word segmentation is not the target of the evaluation. For example, the two parses in the example below only differ in the segmentation of 全球(“the whole world”). However, when the two parse trees are compared, the SParseval metric returns a recall (precision as well since there is the same number of constituents in the two parse trees) of $11/13 = 0.84$. Apparently, both layers of NP that dominate this character sequence are considered to be wrong by the SParseval metric due to the different segmentations of this character string. The SParseval seems to be overly harsh, especially considering that there are cases in Chinese where reasonable people differ as to what the correct word segmentations are.

To address this problem, we developed an alternative *C*Parseval metric that assumes *a constituent is correctly parsed if it spans over the same character sequence as a constituent in the gold standard parse*. This metric factors out word segmentation errors when evaluating syntactic parses. With this metric, there is a complete match between (a) and (b), below.

- a. (IP (NP-SBJ (NP (NN全球)
 (QP (OD 第五)
 (CLP (M 个)))
 (NP-PN (NR 迪斯尼)
 (NN 乐园)))
 (VP (ADVP (AD 即将)
 (PP-LOC (P 在)
 (NP (PN 这里)))
 (PP-DIR(P 向)
 (NP (NN 公众)))
 (VP (VV 开放)))
 (PU 。))
- b. (IP (NP-SBJ (NP (NN全)(NN球)
 (QP (OD 第五)
 (CLP (M 个)))
 (NP-PN(NR迪斯尼)
 (NN乐园)))
 (VP (ADVP (AD 即将)
 (PP-LOC (P 在)
 (NP (PN 这里)))
 (PP-DIR(P 向)
 (NP (NN 公众)))
 (VP (VV 开放)))
 (PU 。))

Translation: The fifth Disney World around the globe is going to be open here to the public soon.

1.4.4.5. Experimental Results

With the evaluation metrics in place, we are ready to report parsing evaluation results. The parsing results are reported for the Bikel parser retrained on the training set and tested on the three test sets described in Section 1.4.4.2. The accuracy is reported in terms of labeled F1 measure and all nodes in the tree are counted, including unary nodes. There are three points of comparison in the results presented Table 1.24. The results are for all three sources (Xinhua, Sinorama and Broadcast news) of CTB 6.0, and the parser takes both gold standard and automatic word segmentations as input, and the parser output is evaluated against both SParseval and CParseval metrics.

When gold standard word segmentation is used as input, the CParseval metric consistently returns slightly lower parsing accuracy, indicating CParseval is not generally a more lenient metric than SParseval. However, when automatic word segmentation is used as input to the parser, the accuracy returned by the CParseval metric is consistently higher, indicating a significant number of errors reported with the SParseval metric are due to incorrect word segmentation. The constituents built on top of wrongly segmented words may not be incorrect in that they still span over the correct character strings.

Genre	Seg	SParseval (f1)	CParseval (f1)
Xinhua	gold	80.37	79.52
Xinhua	auto	75.93	77.13
Sinorama	gold	71.37	70.20
Sinorama	auto	63.41	65.96
BN	gold	79.39	78.51
BN	auto	74.56	76.44

Table 1.24: Comparing SParseval and CParseval.

The third point to notice is that Sinorama data is substantially harder to parse than the other two sources, and there is also a larger divide between the parsing accuracy for gold standard and automatic word segmentations and presumably this is due to the lower word segmentation accuracy for this data set. Over-all, the parsing performance degradation caused by using automatic word segmentation ranges from 2% to 4% using the CParseval metric, and from 4% to 8% using the SParseval metric.

Table 1.25 presents results with unary nodes included or excluded, using just the CParseval metric. When unary nodes are excluded from calculation, the accuracy is consistently higher for all three sources and for both experimental conditions (when gold stan-

dard word segmentation or automatic word segmentation is used as input to the Bikel parser). It is also interesting to note that the difference is around 2% by F1 measure for all experimental conditions.

Genre	Seg	F1 (w/o unary)	F1 (w/ unary)
Xinhua	gold	81.47	79.52
Xinhua	auto	79.04	77.13
Sinorama	gold	72.72	70.20
Sinorama	auto	68.36	65.96
BN	gold	80.60	78.51
BN	auto	78.47	76.44

Table 1.25: CParseval labeled accuracy with and without unary nodes.

1.4.4.6. Conclusion

This section reports an attempt to formally evaluate the impact of automatic word segmentation on syntactic parsing. Given a state-of-the-art word segmenter, the degradation in parsing performance when using the automatic word segmentation as input ranges from 4% to 8% in F1 score by the SParseval metric, depending on the source. We also described the CParseval metric that factors out the word segmentation errors from the parsing accuracy calculation. By this metric the parsing accuracy degradation is 2% to 4% in F1 score. The substantial drop in accuracy floated around informally in the GALE community is likely a result of lower word segmentation accuracy and overly harsh evaluation metrics.

1.4.5. Combining Discriminative Re-ranking and Co-training for Parsing Mandarin Speech Transcripts

Author: Wen Wang

1.4.5.1. Introduction

Discriminative re-ranking has significantly improved parsing performance, and co-training has proven to be an effective weakly supervised learning algorithm to bootstrap parsers from a small in-domain seed labeled corpus using a large amount of unlabeled in-domain data. In this section, we present systematic investigations on combining discriminative re-ranking and co-training, including co-training reranked parsers and co-training rerankers. We show that combining discriminative re-ranking and co-training could improve the F-measure by 1.8%–2% absolute compared to co-training two state-of-the-art Chinese parsers without re-ranking for parsing Mandarin broadcast news and conversation transcripts.

Parsing aims at resolving structural ambiguity. State-of-the-art statistical parsers require treebanks to estimate their parameters, but their performance degrades when there is mismatch on genres/domains between the training treebank and the data to parse. Furthermore, creating a high-quality in-genre/in-domain treebank for the data to parse is ex-

pensive and difficult. However, under the GALE program, there are new genres besides newswire text, namely, broadcast news (BN), broadcast conversation (BC), newsgroup (NG), and web log (WL). Generating high-quality parse trees for Chinese data in these genres can be useful for various tasks within GALE, including syntax-guided translation and reordering models for Chinese-to-English machine translation (MT), name entity detection, and structured language modeling for automatic speech recognition (ASR) on Mandarin BN and BC audio.

In our earlier research (Wang 2008), we employed the weakly supervised co-training technique on two state-of-the-art parsers, Charniak's parser and the Berkeley parser, to bootstrap them from a newswire Chinese treebank and a small amount of BN and BC seed annotated treebank with a large amount of unlabeled BN and BC transcripts, in order to achieve high parsing accuracy on Mandarin BN and BC transcripts. By employing co-training, we obtained 2.2% – 2.6% absolute improvement on F-measure for parsing BN and BC transcripts. F-measure is based on labeled Precision (LP) and labeled Recall (LR). LP is the number of correct constituents divided by the number of constituents found by the parser, and LR is the number of correct constituents divided by the number of constituents in the gold parse. F-measure is defined as $F1 = \frac{2PR}{P+R}$. On the other hand, discriminative re-ranking for parsers (Collins and Koo 2005; Charniak and Johnson 2005) has produced significant improvement on parsing accuracy. In this section, we explore the effectiveness of combining discriminative re-ranking and co-training to further improve parsing performance on Mandarin BN and BC transcripts.

1.4.5.2. Discriminative Re-ranking

We first describe our use of the RankBoost-based discriminative re-ranking approach that was originally developed by Collins and Koo (Collins and Koo 2005) for parsing. This approach allows us to investigate the impact of various features on Mandarin parsing performance. The re-ranking algorithm takes as input a list of candidates produced by a Chinese parser and reranks these candidates based on a set of features. For training the reranker for the parsing task, there are n sentences $\{s_i : i = 1, \dots, n\}$ each with n_i candidates $\{x_{i,j} = 1, \dots, n_i\}$ along with the log-probability $L(x_{i,j})$ produced by the parser. Each parsing candidate $x_{i,j}$ in the training data has a score $Score(x_{i,j})$ that measures the similarity between the candidate and the gold reference. For parsing, we use parse accuracy as the similarity measure. Without loss of generality, we assume that $x_{i,1}$ has the highest score, i.e. $Score(x_{i,1}) \geq Score(x_{i,j})$ for $j = 2, \dots, n_i$. A set of indicator functions $\{h_k : k = 1, \dots, m\}$ is used to extract binary features, $\{h_k(x_{i,j}) : k = 1, \dots, m\}$ on each example $x_{i,j}$. Each indicator function h_k is associated with a weight parameter α_k that is real valued. In addition, a weight parameter α_0 is associated with the log-probability $L(x_{i,j})$. The ranking function of candidates $x_{i,j}$ is defined as $\alpha_0 L(x_{i,j}) + \sum_{k=1}^m \alpha_k h_k(x_{i,j})$. The objective of the training process is to set the parameters $\bar{\alpha} = \{\alpha_0, \alpha_1, \dots, \alpha_m\}$ to minimize the loss function $Loss(\bar{\alpha})$ (which is an upper bound on the training error), as $Loss(\bar{\alpha}) = \sum_i \sum_{j=2}^{n_i} S_{i,j} e^{-M_{i,j}(\bar{\alpha})}$ where $S_{i,j}$ is the weight function that gives the importance of each example, and $M_{i,j}(\bar{\alpha})$ is the margin (Collins

and Koo 2005). All the α_i 's are initially set to zero. Then a greedy sequential optimization method is used in each boosting round to select the feature that has the most impact on reducing the loss function and then update its weight parameter accordingly.

Collins' method allows multiple updates to the weight of a feature. Huang *et al.* (2007) found that for those strong features, Collins' weight update formula can increase their weight (in absolute value) in only one direction. Although these features are strong and useful, setting weights too large can be harmful in that it limits the use of other features for reducing the loss. Based on this analysis, Huang *et al.* (2007) have developed an update-once method, in which the weight update is limited so that once a feature is selected in a certain iteration and its weight parameter is updated, no update will be conducted on it again. In this way, the weights of the strong features will not be allowed to prevent other features from being considered during the training procedure. Huang *et al.* observed that the update-once method could select significantly more features compared to Collins' original method and produce better reranking performance. In this section, we employed this update-once strategy for updating feature weights

For the work described in this section, we employed the features described by Collins and Koo (2005). Note that before generating these features, we applied headword percolation on the trees that are the output of parsers, as employed by Collins and Koo (2005). Features include rules (all context-free rules in the tree), bigrams (adjacent pairs of non-terminals to the left and right of the head), grand-parent rules (same as rules, but also including the non-terminal above the rule), head-modifiers (all head-modifier pairs, also including the grandparent non-terminal), and PPs (lexical trigrams involving the heads of arguments of prepositional phrases) and so on. More details are presented by Collins and Koo (2005).

1.4.5.3. Co-training

Co-training was first introduced by Blum and Mitchell (Blum and Mitchell 1998) as a weakly supervised learning method and can be used for bootstrapping a model from a seed corpus of labeled examples, which is typically quite small, augmented with a much larger amount of unlabeled examples, by exploiting redundancy among multiple statistical models that generate different *views* of the data. Informally, co-training can be described as picking multiple classifiers (“views”) of a classification problem, building models for each view and training these models on a small set of labeled data, then on a large set of unlabeled data, sampling a subset, labeling the sampled subset using the models, selecting examples from the labeled results, adding them to the training pool, and iterating this procedure until the unlabeled set is all labeled.

In Figure 1.13, when calling the classifier that provides additional training data for the opposite classifier the *teacher* and the opposite classifier the *student*, since the labeled output from both classifiers h_1 and h_2 is noisy, an important question is which newly labeled examples from the teacher should be added to the training data pool of the student. This issue of example selection plays an important role in the learning rate of co-training and the performance of resulting classifiers. In previous work, investigated four example selection approaches, namely, *naive co-training*, *agreement-based co-training*, *max-score*, and *max-t-min-s* (Wang 2008). We also developed the *max-score* and *max-t-min-s*

approaches (Wang *et al.* 2007). We also compare the performance of co-training to self-training. Self-training in this work simply adds all examples in the labeled cache to the training pool in each iteration (Nigram and Ghani 2000).

Input: S is a seed set of labeled data.
 L_{h_1} is labeled training data for h_1 .
 L_{h_2} is labeled training data for h_2 .
 U is the unlabeled data set.
 C is the cache holding a small subset of U .

- 1 $L_{h_1} \leftarrow S$
- 2 $L_{h_2} \leftarrow S$
- 3 Train classifier h_1 on L_{h_1}
- 4 Train classifier h_2 on L_{h_2}
- 5 **repeat**
- 6 Randomly partition U into C where $|C| = u$ and U'
- 7 Apply h_1, h_2 to assign labels for all examples in C
- 8 Select examples labeled by h_1 and add to L_{h_1}
- 9 Train h_2 on L_{h_2}
- 10 Select examples labeled by h_2 and add to L_{h_1}
- 11 Train h_1 on L_{h_1}
- 12 $U \leftarrow U'$
- 13 **until** U is empty

Figure 1.13: General co-training algorithm.

In previous work (Wang 2008), we systematically investigated applying weakly supervised co-training approaches to improve parsing performance for parsing Mandarin BN and BC transcripts, by iteratively retraining two competitive Chinese parsers, Charniak's re-ranking parser (Charniak and Johnson 2005) and the Berkeley parser (Petrov *et al.* 2006), from a small set of treebanked data and a large set of un-labeled data. Compared to parsers trained only on the small in-domain seed labeled corpus, the parsers resulting from co-training could gain 6.8% absolute on BN and 7.3% absolute on BC based on the F-measure.

Overall, compared to parsers trained on all available treebank data including in-domain and out-of-domain treebanks, co-training yields a 2.2%–2.6% absolute gain on BN and 2.4%–2.5% absolute gain on BC based on the F-measure (and 1.5%–1.9% absolute gain on BN and 1.7%–2.0% absolute gain on BC over self-training (Wang 2008)). In this section, we investigate the combination of discriminative re-ranking and co-training on Charniak's maximum-entropy inspired parser (Charniak 2000) (i.e., the parser without re-ranking compared to the parser of Charniak and Johnson (2005)) and the Berkeley parser (also originally without re-ranking). For co-training parsers, we employed the *max-t-min-s* example selection approach, as it is computation-ally inexpensive and also produced the best performance (Wang 2008).

1.4.5.4. Combining Re-ranking and Co-training

Both Charniak and Berkeley parsers support generating N-best parses for re-ranking purposes. In fact, Charniak and Johnson have implemented a discriminative reranker using a MaxEnt estimator to find the feature weights and when using the reranker to rerank 50-best parses from Charniak's Maximum-entropy inspired parser, it improved F-measure by 1.3% absolute on sentences of length less than 100 words in Wall Street Journal Penn Treebank section 23 (Charniak and Johnson 2005).

In this work, we adopted this reranker for Charniak's parser, implemented the RankBoost-based re-ranking algorithm described in Section 1.4.5.2 to rerank 50-best from the Berkeley parser, and then investigated two ways to combine re-ranking and co-training. The direct combination approach is for each iteration of co-training, instead of generating 1-best parse directly from the no-re-ranking, standard Charniak and Berkeley parsers, 50-best parses are generated from each parser and then reordered by their corresponding rerankers, respectively. Then the 1-best parses after re-ranking for the unlabeled data are selected and added to the training pool of the parsers. In this section this approach is denoted **co-training reranked parsers**. Note that for this approach the features and feature weight parameters for rerankers remain the same during the co-training procedure.

Different from the original binary classification problems on which co-training was developed, parsing contains a number of smaller decisions about which constituents are probable, and inherently each parser includes good and bad decisions on how to create/attach different constituents. On the other hand, re-ranking is closer to binary classification than parsing, as it tries to decide whether or not a parse hypothesis is the best parse for the sentence, so it is explicit to maximize agreement between rerankers, as the principled agreement-based example selection approach could be applied here, which could guarantee co-training to improve parsing accuracy. Hence, we hypothesize that co-training rerankers could better fit the co-training algorithm. For effectiveness, rerankers can consider features that span the entire tree of a parse (while parsers generally consider only local features). For efficiency, co-training rerankers requests unlabeled data to be parsed just once, compared to multiple parsing iterations for co-training reranked parsers. The output will be reranked many times, but this is much more efficient than training and running parsers. Hence, in this work, we also investigated co-training our RankBoost-based reranker with Charniak's and Johnson's MaxEnt reranker and applied the co-trained rerankers to the two standard parsers. This second approach is denoted **co-training rerankers**.

1.4.5.5. Data

For selecting parsers and also contributing to training parsers, we used Chinese Treebank 5.2 released by LDC (denoted as CTB). Chinese Treebank 5.2 contains 500K words, 800K characters, 18K sentences, and 900 data files. Under the GALE program, the BN genre follows its tradition and consists of “talking head” style broadcasts, i.e., generally one person reading a news script. The BC genre, by contrast, is more conversational and spontaneous, consisting of talk shows, interviews, call-in programs, and roundtables.

The evaluation of co-training for parsing Mandarin BN and BC transcripts is conducted on the GALE OntoNotes released Mandarin BN and BC treebanks. The BN treebank is from the Mandarin TDT4 collection, and the BC treebank is from GALE Mandarin BC data and translations from English BC data. The Mandarin BN treebank includes 300K words and 814 data files, and the BC treebank 100K words and 16 data files.

To create a seed corpus and a test set for evaluating parsing accuracy, for BN and BC respectively, we divided the whole BN/BC treebank into blocks of 10 files by sorted order. Within each block, the first file is used for co-training development and the second for testing parsing accuracy. The remaining eight files are used as part of the seed annotated corpus for co-training. The resulting BN test set is denoted **BN-test** and the seed annotated corpus **BN-seed**. The BC test set is de-noted **BC-test** and the BC seed annotated corpus **BC-seed**. BN-test includes 31K words and 1,565 sentences. BC-test includes 11K words and 1,482 sentences. The large set of unlabeled data for BN parsing includes Hub4 1997 Mandarin BN acoustic transcripts, LDC Chinese TDT {2,3,4} corpora, Chinese Gigaword 3.0, and all GALE released BN audio transcripts, denoted **BN-unlabeled**. For BC parsing, we add all GALE released BC audio transcripts denoted **BC-unlabeled**. After word segmentation, **BN-unlabeled** comprises around 1.4 billion words and **BC-unlabeled** around 11 million words.

1.4.5.6. Selecting Parsers for Co-training

Parser	F-measure
Charniak	83.2%
Stanford	82.0%
Berkeley	83.5%
Bikel	82.9%

Table 1.26: F-measure of all four parsers on the CTB test set. The train/dev/test split is described in Section 1.4.5.5.

To select the two parsers for co-training, we investigated four publicly available parsers, namely, Charniak's maximum-entropy inspired parser (Charniak 2000), the Stanford unlexicalized parser (Klein and Manning 2003), Berkeley parser (Petrov *et al.* 2006), and Dan Bikel's reimplementation of Michael Collins' Model 2 parser (Collins 1999). To select two from them in our co-training setup, we considered two important factors: accuracy and mutual complementariness. To evaluate parser accuracy, we consider the F-measure. Using the train/dev/test split described in Section 1.4.5.5, Table 1.26 shows the F-measure of all four parsers on the test set.

The co-training principle requires the two views to be conditionally independent or weakly conditionally independent. This means that we need to select parsers that are complementary on their learning patterns and error types. To measure the structural complementariness between parsers, we adapted the measure of structural consistency between parsers and modified the objective function for maximizing the structural complementariness between parsers to be selecting parsers with the minimal structural consistency.

Note that to measure the structural consistency between the bracketing parses from parsers and gold standard parses, Black, Garside, and Leech (Black *et al.* 1993) defined the metric *average crossing brackets* (ACB), the mean number of times per sentence that a bracketed sequence from one parser overlaps with the gold standard from the treebank such that neither is properly contained in the other. Although ACB does not account for all types of conflicting constituency, it is a practical measure for the structural consistency between two sets of parse trees.

By using the output from one parser B as the gold set, we can calculate the pair-wise ACB_{AB} of parser A on parser B . The ACB values on the CTB test set among all six pairs from the four parsers are ordered as {Charniak, Stanford}, 2.11; {Berkeley, Stanford}, 2.09; {Charniak, Bikel}, 2.05; {Berkeley, Bikel}, 2.01; {Charniak, Berkeley}, 1.99; {Bikel, Stanford}, 1.87. Since we need to achieve the best combination of maximizing parsers' accuracy and maximizing their mutual complementariness (i.e., maximizing their pair-wise ACB), we selected Charniak's maximum-entropy inspired parser and the Berkeley parser for co-training.

1.4.5.7. Experimental Results and Discussions

Training Condition	F-measure (%)	
	Charniak	Berkeley
1. BN-seed	75.1	75.2
2. CTB	79.1	79.1
3. CTB+BN-seed	80.4	80.5
4. co-training initialized as Condition 3, max-t-min-s	82.3	82.6

Table 1.27: Overall parsing accuracy F-measure (%) on the Mandarin BN treebank test set, BN-test, after applying co-training using Charniak's maximum-entropy inspired parser and the Berkeley parser, both without re-ranking.

Table 1.27 shows the parsing accuracy F-measure (%) on BN-test under various parser training conditions on Charniak's parser and the Berkeley parser without re-ranking. As can be seen from the table, training Charniak's parser and the Berkeley parser using only the small training set of BN treebank, i.e., BN-seed, resulted in relatively poor parsing performance, at 75.1% F1 for Charniak's parser and 75.2% for the Berkeley parser. Using the larger full CTB corpus for training improves parsing performance significantly and adding BN-seed to CTB brought additional gain. However, co-training using CTB plus BN-seed as the initial training pool significantly improved the performance of the two parsers over directly training on CTB plus BN-seed, with 1.9% absolute and 2.1% absolute improvement on F-measure for Charniak's parser and the Berkeley parser, respectively.

For co-training carried out in these experiments, we used cache size as 10K sentences. Table 1.28 shows the F-measure from the two no-re-ranking parsers on BC-test under various training conditions. The condition **BN-co-trained** denotes the BN-seed treebank and the final annotated BN-unlabeled data after applying *max-t-min-s* co-training to the two parsers initialized on the BN-seed treebank. **BN-co-trained** significantly outperforms **CTB**, indicating greater similarity between the two speech genres

compared to CTB vs. BC. Using CTB and BN-seed to initialize the two parsers and then co-training on the BN-unlabeled data achieved further gain on parsing performance, denoted by **(CTB+BN)-co-trained**. Consistent with Table 1.27 it is always helpful to add the small in-genre seed treebank into training, as **(CTB+BN)-co-trained+BC-seed** outperforms **(CTB+BN)-co-trained**. Co-training on BC-unlabeled also produced consistent improvement on F-measures.

Overall, we gained 2.5% absolute on F-measure on BC-test over the two parsers from co-training. Using the same BC-unlabeled data for co-training, we also compared initializing the two parsers with the condition of CTB only and the condition of adding the small BN-seed and BC-seed corpora, and observed that adding this small in-genre seed corpus always outperforms initializing with CTB only, by 1% on BN and 1.4% on BC.

Training Condition	F-measure (%)	
	Charniak	Berkeley
1. BC-seed	72.0	72.8
2. CTB	73.4	73.7
3. BN-co-trained	74.7	74.8
4. (CTB+BN)-co-trained	75.6	75.7
5. (CTB+BN)-co-trained+BC-seed	76.8	77.0
6. co-training initialized as Condition 5, max-t-min-s	79.3	79.5

Table 1.28: Overall parsing accuracy F-measure (%) on the Mandarin BC treebank test set, BC-test, after applying co-training using Charniak's maximum-entropy inspired parser and the Berkeley parser, both without re-ranking.

Training Condition	BN-test F-measure (%)		BC-test F-measure (%)	
	Charniak	Berkeley	Charniak	Berkeley
co-training standard parsers	82.3	79.3	79.5	82.6
co-training reranked parsers	83.8	81.0	81.1	84.0
co-training rerankers	84.0	81.5	81.5	84.4

Table 1.29: Overall parsing accuracy F-measure (%) on BN-test and BC-test, after applying co-training using Charniak's maximum-entropy inspired parser and the Berkeley parser, both without re-ranking and with re-ranking.

The results from the two approaches of combining discriminative re-ranking and co-training, as proposed in Section 1.4.5.4, are shown in Table 1.29. The results of **co-training standard parsers** are the last rows in Tables 1.27 and 1.28. When co-training reranked parsers, the rerankers were trained on CTB+BN-seed for BN and CTB+BN-seed+BC-seed for BC and remained the same during co-training. When co-training rerankers, the rerankers were initialized on CTB+BN-seed for BN and CTB+BN-seed+BC-seed for BC and updated during co-training. For both combination approaches, co-training explored BN-unlabeled for BN and BC-unlabeled for BC as unlabeled data, respectively.

As can be seen, **co-training reranked parsers** (using the *max-t-min-s* example selection approach) significantly outperforms co-training without re-ranking, by 1.5% absolute

and 1.4% absolute gain on F-measure on the two parsers on BN-test, and 1.7% absolute and 1.6% absolute gain on F-measure on the two parsers on BC-test. For **co-training rerankers**, as discussed in Section 1.4.5.4, it is feasible now for us to employ the more principled agreement-based example selection approach during co-training since we can simply train each reranker multiple times on different subsets of the automatically labeled data and examine which partition of the data produced the maximum agreement among the rerankers.

As a reminder, for **co-training reranked parsers**, we still used the *max-t-min-s* approach as it is computationally efficient and also proved to be very effective for co-training parsers (Wang 2008). As can be seen from the tables, **co-training rerankers** produced a small yet consistent gain over **co-training reranked parsers**, by 0.2% – 0.4% absolute improvement on BN-test and 0.4% – 0.5% absolute improvement on BC-test, raising the absolute improvement on F-measure up to 1.8% on BN-test and 2% on BC-test, from combining discriminative re-ranking and co-training compared to co-training only.

References

- Al-Sughaiyer, I. A. and I. A. Al-Kharashi. 2004. Arabic Morphological Analysis Techniques: A Comprehensive Survey. *Journal of the American Society for Information Science and Technology* 55(3):189–213.
- Babko-Malaya, O. 2008. Annotation of Nuggets and Relevance in GALE Distillation Evaluation. *Proceedings of the International Conference on Language Resources and Evaluation*. Marrakech, Morocco.
- Babko-Malaya, O., A. Bies, A. Taylor, S. Yi, M. Palmer, M. Marcus, S. Kulick and L. Shen. 2006. Issues in Synchronizing the English Treebank and PropBank. *Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora*.
- Babko-Malaya, O., M. Palmer, N. Xue, A. Joshi and S. Kulick. 2004. Proposition Bank II: Delving Deeper. *Proceedings of the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics Frontiers in Corpus Annotation Workshop*.
- Babko-Malaya, O., Z. Song, R. Zakhary, S. Strassel and J. Medero. 2006. GALE Y1 - Distillation Training Query Answer Keys V7.0. LDC2006E15.
- Badr, I., R. Zbib and J. Glass. 2008. Segmentation For English-To-Arabic Statistical Machine Translation. *Proceedings of the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics*. Short Papers: 153–156. Columbus, Ohio.
- Baker, C.F., C. J. Fillmore and J. B. Lowe. 1998. The Berkeley FrameNet Project. *Proceedings of the International Conference on Computational Linguistics/Meeting of the Association for Computational Linguistics* pp. 86-90.
- Benajiba, Y., M. Diab and P. Rosso. 2008. Arabic Named Entity Recognition Using Optimized Feature Sets. *Proceedings of the 2008 Conference on Empirical Methods on Natural Language Processing* pp. 284–293. Honolulu, Hawaii.
- Benajiba, Y., M. Diab and P. Rosso. 2009. Arabic Named Entity Recognition: A Feature-Driven Study. *IEEE Transactions on Audio, Speech & Language Processing* 17(5): 926-934.
- Bies, A. and M. Ferguson, K. Katz and R. MacIntyre, Eds. 1995. Bracketing Guidelines for Treebank II Style. *Penn Treebank Project, University of Pennsylvania, CIS Technical Report MS-CIS-95-06*.
- Bikel, D.M. 2004. On the Parameter Space of Generative Lexicalized Statistical Parsing Models. *Ph.D. thesis, University of Pennsylvania*.
- Bikel, D.M. and D. Chiang. 2000. Two Statistical Parsing Models Applied To The Chinese Treebank. *Proceedings of the Second Chinese Language Processing Workshop*.
- Black, E., R. Garside and G. Leech. 1993. *Statistically-driven Computer Grammars of English: The IBM/Lancaster Approach*. Rodopi, Amsterdam.

- Blum, A. and T. Mitchell. 1998. Combining Labeled and Unlabeled Data With Co-Training. *Proceedings of the Conference on Learning Theory*.
- Buckwalter, T. 2002. Buckwalter Arabic Morphological Analyzer Version 1.0. *Linguistic Data Consortium, University of Pennsylvania*. LDC Catalog No.: LDC2002L49.
- Buckwalter, T. 2004. Buckwalter Arabic Morphological Analyzer Version 2.0. *Linguistic Data Consortium, University of Pennsylvania*. LDC Catalog No.: LDC2004L02.
- Burchardt, A., K. Erk, A. Frank, A. Kowalski, S. Pado and M. Pinkal. 2006. Consistency And Coverage: Challenges for Exhaustive Semantic Annotation. *Proceedings of The German Society for Linguistics Conference (DGfS)*.
- Chang, P.C., M. Gally and C. Manning. 2008. Optimizing Chinese Word Segmentation for Machine Translation Performance. *Proceedings of the Meeting of the Association for Computational Linguistics Third Workshop on Statistical Machine Translation*.
- Charniak, E. 1997. Statistical Parsing with a Context-Free Grammar and Word Statistics. *Proceedings of the International Conference on Artificial Intelligence*.
- Charniak, E. 2000. A Maximum-Entropy-Inspired Parser. *Proceedings of the Meeting of the North American Chapter of the Association for Computational Linguistics*.
- Charniak, E. and M. Johnson. 2005. Coarse-to-fine Nbest Parsing and MaxEnt Discriminative Reranking. *Proceedings of the 43rd Meeting of the Association for Computational Linguistics*.
- Chen, J. and M. Palmer. 2005. Towards Robust High Performance Word Sense Disambiguation of English Verbs Using Rich Linguistic Features. *Proceedings of the International Joint Conference on Natural Language Processing* pp. 933-944.
- Chen, K.J. and S. H. Liu. 1992. Word Identification for Mandarin Chinese Sentences. *Proceedings of the International Conference on Computational Linguistics* pp. 101-107.
- Cieri, C., S. Strassel, D. Graff, N. Martey, Ka. Rennert and M. Liberman. 2002. Corpora for topic detection and tracking. James Allen, editor, *Topic Detection and Tracking: Event-based Information Organization*. Kluwer Academic Publishers.
- Cieri, C. and Liberman, M. 2000. Issues in Corpus Creation and Distribution: The Evolution of the Linguistic Data Consortium. *Proceedings of the International Conference on Language Resources and Evaluation*.
- Cieri, C., S. Strassel, M. Glenn and L. Friedman. 2007. Linguistic Resources in Support of Various Evaluation Metrics. *Automatic Procedures in MT Evaluation Workshop: MT Summit XI*.
- Collins, M. 1999. Head-Driven Statistical Models for Natural Language Parsing. *Ph.D. thesis, University of Pennsylvania*.
- Collins, M. and T. Koo. 2005. Discriminative Reranking For Natural Language Parsing. *Computational Linguistics*, 31(1):25-70.

Costa-jussà, M.R., J.M. Crego, A. de Gispert, P. Lambert, M. Khalilov, J.A.R. Fonollosa, J.B. Mariño and R. Banchs. 2006. TALP Phrase-Based System and TALP System Combination. *Proceedings of the International Workshop on Spoken Language Translation* pp. 123–129. Kyoto, Japan.

Cowan, M. 2009. *GNU Wget*. <http://www.gnu.org/software/wget>.

Crego, J.M. and N. Habash. 2008. Using Shallow Syntax Information to Improve Word Alignment and Reordering for SMT. *Proceedings of the Third Workshop on Statistical Machine Translation* pp. 53–61. Columbus, Ohio.

CRL Lesser Studied Languages Center. 2009. <http://crl.nmsu.edu/say/>

Dean, J. and S. Ghemawat. 2005. MapReduce: Simplified Data Processing on Large Clusters. *Proceedings of the Sixth Symposium on Operating System Design and Implementation (OSDI'04)*.

Diab, M. 2007. Improved Arabic Base Phrase Chunking with a New Enriched Pos Tag Set. *Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources* pp. 89–96. Prague, Czech Republic.

Diab, M. 2007a. Towards An Optimal Pos Tag Set For Modern Standard Arabic Processing. *Proceedings of the Conference on Recent Advances in Natural Language Processing*. Borovets, Bulgaria.

Diab, M., K. Hacioglu and D. Jurafsky. 2004. Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. *Proceedings of the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics*. Boston, MA.

Diab, M., M. Alkhalifa, S. ElKateb, C. Fellbaum, A. Mansouri, M. Palmer. 2007. SemEval-2007 Task 18: Arabic Semantic Labeling. *Proceedings of the International Workshop on Semantic Evaluations*.

Diab, M., M. Ghoneim and N. Habash. 2007a. Arabic Diacritization In The Context Of Statistical Machine Translation. *Proceedings of the Machine Translation Summit*. Copenhagen, Denmark.

Diab, M., K. Hacioglu and D. Jurafsky. 2007b. Automatic Processing of Modern Standard Arabic Text. *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Abdelhadi Soudi, Antal van den Bosch and Günter Neumann, eds. Springer.

Dligach, D. and M. Palmer. 2008. Novel Semantic Features for Verb Sense Disambiguation. *Proceedings of the Meeting of the Association for Computational Linguistics*.

Dligach, D. and M. Palmer. 2009. Using Language Modeling to Select Useful Annotation Data. *Proceedings of the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics*. Boulder, Colorado.

Elming, J. and N. Habash. 2007. Combination of Statistical Word Alignments Based on Multiple Preprocessing Schemes. *Proceedings of the Human Language Technologies*

Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics Companion Volume, Short Papers: 25–28. Rochester, New York.

Elming, J., N. Habash and J. Crego. 2008. Combination Of Statistical Word Alignments Based On Multiple Preprocessing Schemes. Cyril Goutte, Nicola Cancedda, Marc Dymetman and George Foster, eds., *Learning for Machine Translation*. MIT Press.

Emerson, T. 2005. The Second International Chinese Word Segmentation Bakeoff. *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* pp. 123–133. Jeju Island, Korea.

Farber, B., D. Freitag, N. Habash and O. Rambow. 2008. Improving NER in Arabic Using a Morphological Tagger. *Proceedings of the International Conference on Language Resources and Evaluation*. Marrakech, Morocco.

Farwell, D., J. Giménez, E. González, R. Halkoum, H. Rodríguez and M. Surdeanu. 2007. The UPC System for Arabic-to-English Entity Translation. *Proceedings of the Automatic Content Extraction Conference*.

Favre, B., D. Hakkani-Tur, S. Petrov and D. Klein. 2008. Efficient Sentence Segmentation Using Syntactic Features. *Proceedings of the IEEE/Meeting of the Association for Computational Linguistics Workshop on Spoken Language Technologies*.

Favre, B., R. Grishman, D. Hillard, H. Ji, D. Hakkani-Tur and M. Ostendorf. 2008a. Punctuating Speech for Information Extraction. *Proceedings of the International Conference on Acoustics, Speech and Signal Processing* pp. 5013–5106.

Fellbaum, C. ed. 1998. WordNet: An On-line Lexical Database and Some of its Applications. *MIT Press*.

Fiscus, J., J. Ajot, N. Radde and C. Laprun. 2006. Multiple Dimension Levenshtein Edit Distance Calculations for Evaluating Automatic Speech Recognition Systems During Simultaneous Speech. *Proceedings of the International Conference on Linguistic Resources and Evaluation*.

Fraser, A. and D. Marcu. 2007. Measuring Word Alignment Quality for Statistical Machine Translation. *Computational Linguistics* 33(3):293-303

Friedman, L. and S. Strassel. 2008. Identifying Common Challenges for Human and Machine Translation: A Case Study from the GALE Program. *Proceedings of the Meeting of the Association for Machine Translation in the Americas*. Waikiki, Hawaii.

Friedman, L., H. Lee and S. Strassel. 2008. A Quality Control Framework for Gold Standard Reference Translations: The Process and Toolkit Developed for GALE. *Translating and the Computer 30 Proceedings*. London, UK.

Friedman, L., S. Strassel and M. Glenn. 2008. Explicit and Implicit Requirements of Technology Evaluations: Implications for Test Data Creation. *Proceedings of the International Conference on Language Resources and Evaluation*. Marrakech, Morocco.

Gabbard, R. and S. Kulick. 2008. Construct State Modification in the Arabic Treebank. *Proceedings of Meeting of the Association for Computational Linguistics Short Papers*.

- Gao, J.F., M. Li, A. Wu and C.N. Huang. 2005. Chinese Word Segmentation and Named Entity Recognition: a Pragmatic Approach. *Computational Linguistics*, 31(4):531–574.
- Glenn, M. and S. Strassel. 2006. Shared Linguistic Resources for the Meeting Domain. *Proceedings of the Classification of Events, Activities, and Relationships Evaluation and RT Evaluation*.
- Glenn, Meghan Lammie, Haejoong Lee and Stephanie M. Strassel. 2009. XTrans: a speech annotation and transcription tool. *Proceedings of Interspeech 2009, Brighton, UK*.
- Habash, N. 2004. Large Scale Lexeme Based Arabic Morphological Generation. *Proceedings of the Conference on Automated Processing of Natural Languages (TALN-04)* pp. 271–276. Fez, Morocco.
- Habash, N. 2007. Arabic Morphological Representations for Machine Translation. *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Abdelhadi Souidi, Antal van den Bosch and Günter Neumann, eds. Springer.
- Habash, N. and F. Sadat. 2006. Arabic Preprocessing Schemes for Statistical Machine Translation. *Proceedings of the 7th Meeting of the North American Chapter of the Association for Computational Linguistics/ Human Language Technologies Conference* pp. 49–52. New York, New York.
- Habash, N. and O. Rambow. 2005. Arabic Tokenization, Morphological Analysis, and Part-of-Speech Tagging in One Fell Swoop. *Proceedings of the Meeting of the Association for Computational Linguistics*. Ann Arbor, Michigan.
- Habash, N. and O. Rambow. 2006. Magead: A morphological analyzer for Arabic and its dialects. *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*. Sydney, Australia.
- Habash, N. and O. Rambow. 2007. Arabic Diacritization through Full Morphological Tagging. *Proceedings of the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics*.
- Habash, N. and R. Roth. 2008. Identification of Naturally Occurring Numerical Expressions in Arabic. *Proceedings of the International Conference on Language Resources and Evaluation*. Marrakech, Morocco.
- Habash, N. and R. Roth. 2009. CATiB: The Columbia Arabic Treebank. *Proceedings of the Conference of the Association for Computational Linguistics*. Suntec, Singapore.
- Habash, N., O. Rambow and G. Kiraz. 2005. Morphological Analysis and Generation for Arabic Dialects. *Proceedings of the Meeting of the Association for Computational Linguistics, Workshop on Computational Approaches to Semitic Languages*. Ann Arbor, MI.
- Habash, N., A. Souidi and T. Buckwalter. 2007. On Arabic Transliteration. *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Abdelhadi Souidi, Antal van den Bosch, and Günter Neumann, eds. Springer.

- Habash, N., R. Faraj and R. Roth. 2009. Syntactic Annotation in the Columbia Arabic Treebank. *Proceedings of the International Conference on Arabic Language Resources and Tools (MEDAR)*. Cairo, Egypt.
- Hajic, A.J., B. Vidová-Hladká and P. Pajas. 2001. The Prague Dependency Treebank: Annotation Structure and Support. *Proceeding of the IRCS Workshop on Linguistic Databases*, pp. 105–114.
- Hajič, J., O. Smrž, P. Zemánek, J. Šnaidauf and E. Beška. 2004. Prague Arabic Dependency Treebank: Development in Data and Tools. *Proceedings of the Network for Euro-Mediterranean Language Resources Conference on Arabic Language Resources and Tools*. Cairo, Egypt.
- Harper, M., B. Dorr, J. Hale, B. Roark, I. Shafran, M. Lease, Y. Liu, M. Snover, L. Yung, A. Krasnyanskaya and R. Stewart. 2005. *The Johns Hopkins Summer Workshop Final Report on Parsing and Spoken Structural Event Detection*. Johns Hopkins University.
- Hillard, D., Z. Huang, H. Ji, R. Grishman, D. Hakkani-Tur, M. Harper, M. Ostendorf and W. Wang. 2006. Impact of Automatic Comma Prediction on POS/Name Tagging Of Speech. *Proceedings of the IEEE/Meeting of the Association for Computational Linguistics Workshop Spoken Language Technology* pp. 58–61.
- Holovaty, A. 2007. Templatemaker. <http://code.google.com/p/templatemaker/>.
- Hovy, E.H., A. Philpot, *et al.* 2009. The Omega Upper Model. Unpublished ms.
- Huang, C.R., T.S. You, P. Simon and S.K. Hsieh. 2008. A Realistic and Robust Model for Chinese Word Segmentation. *Proceedings of the Conference on Computational Linguistics and Speech Processing (ROCLING-2008)*. Taiwan.
- Huang, J. and G. Zweig. 2002. Maximum Entropy Model for Punctuation Annotation from Speech. *Proceedings of the International Conference on Spoken Language Processing* pp.917–920.
- Huang, Z., M. Harper and W. Wang. 2007. Mandarin Part-of-speech Tagging and Discriminative Reranking. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*.
- Hwang, M.Y., X. Lei, W. Wang and T. Shinozaki. 2006. Investigation on Mandarin Broadcast News Speech Recognition. *Proceedings of the International Conference on Spoken Language Processing*.
- Jin, G. and X. Chen. 2008. The Fourth International Chinese Word Segmentation Bakeoff. *Proceedings of the Sixth SIGHAN Workshop on Chinese Language Processing*. Hyderabad, India.
- Jones, D., E. Gibson, W. Shen, N. Granoien, M. Herzog, D. Reynolds and C. Weinstein. 2005. Measuring Human Readability of Machine-Generated Text: Three Case Studies in Speech Recognition and Machine Translation. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing* pp. 1009–1012.

- Kahn, J.G., M. Ostendorf and C. Chelba. 2004. Parsing Conversational Speech Using Enhanced Segmentation. *Proceedings of the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics* pp. 125–128.
- Kiraz, G.A. 2000. Multi-tiered Nonlinear Morphology Using Multi-tape Finite Automata: A Case Study on Syriac and Arabic. *Computational Linguistics* 26(1):77–105.
- Klein, D. and C. Manning. 2003. Accurate Unlexicalized Parsing. *Proceedings of the 41st Meeting of the Association for Computational Linguistics* pp. 423–430.
- Kruijff-Korbyová, I., K. Chvátalová, O. Postolache. 2006. Annotation Guidelines for Czech-English Word Alignment. *Proceedings of the International Conference on Language Resources and Evaluation* pp. 1256–1261.
- Kulick, S., R. Gabbard and M. Marcus. 2006. Parsing the Arabic Treebank: Analysis and Improvements. *Proceedings of the Treebanks and Linguistic Theories Conference*. Prague, Czech Republic.
- Larkey, L.S., L. Ballesteros and M.E. Connell. 2007. Light Stemming for Arabic Information Retrieval. *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Abdelhadi Souidi, Antal van den Bosch, and Günter Neumann, eds. Springer.
- Le, S., Y. Jin, L. Du and Y. Sun. 2000. Word Alignment of English-Chinese Bilingual Corpus Based on Chunks. *Proceedings of the 2000 Joint SIGDAT conference on Empirical Methods on Natural Language Processing and Very Large Corpora/ The 38th Annual Meeting of the Association for Computational Linguistics* 13: 110 – 116.
- Levow, G.A. 2006. The Third International Chinese Word Segmentation Bakeoff: Word Segmentation and Named Entity Recognition. *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*. Sydney, Australia.
- Levy, R. and C. Manning. 2003. Is it Harder to Parse Chinese, or the Chinese Treebank. *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*. Sapporo, Japan.
- Levy, R. and G. Andrew. 2006. Tregex and Tsurgeon: Tools for Querying and Manipulating Tree Data Structures. *Proceedings of the International Conference on Language Resources and Evaluation*.
- LDC. 2004. *Simple Metadata Annotation Specification, Version 6.2*.
http://projects ldc.upenn.edu/MDE/Guidelines/SimpleMDE_V6.2.pdf
- LDC. 2006. *Simple Named Entity Guidelines for Less Commonly Taught Languages, Version 6.5*.
<http://projects ldc.upenn.edu/LCTL/Specifications/SimpleNamedEntityGuidelinesV6.5.pdf>
- LDC. 2007. *Using XTrans for Broadcast Transcription: A User Manual, Version 3.0*.
<http://projects ldc.upenn.edu/gale/Transcription/XTransManualV3.pdf>

- LDC. 2008. *GALE P3 Supra-Lexical Annotation Experiment Guidelines*. http://projects.ldc.upenn.edu/gale/Transcription/Supra-lexicalAnnotation_v0.6.pdf
- LDC. 2009. *GALE Website*. <http://projects.ldc.upenn.edu/gale/index.html>
- Liu, F. and Y. Liu. 2007. Soundbite Identification Using Reference and Automatic Transcripts of Broadcast News Speech. *Proceedings of the Automatic Speech Recognition and Understanding Workshop* pp. 653–658.
- Liu, Y., E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf and M. Harper. 2006. Enriching Speech Recognition with Automatic Detection of Sentence Boundaries and Disfluencies. *IEEE Transactions on Audio, Speech, and Language Processing* 14(5):1526–1540.
- Low, J.K., H.T. Ng and W.Y. Guo. 2005. A Maximum Entropy Approach to Chinese-Word Segmentation. *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing* pp. 161–164. Jeju Island, Korea.
- Luo, X.Q. 2003. A Maximum Entropy Chinese Character-Based Parser. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*. Sapporo, Japan.
- Ma, X. 2006. Champollion: A Robust Parallel Text Sentence Aligner. *Proceedings of LREC-2006*, Genoa, Italy.
- Ma, X. and M. Liberman. 1999. BITS: A method for bilingual text search over the web. *Proceedings of the Machine Translation Summit VII*.
- Maamouri, M. and A. Bies. 2004. Developing an Arabic Treebank: Methods, Guidelines, Procedures, and Tools. *Proceedings of the International Conference on Computational Linguistics*. Geneva, Switzerland.
- Maamouri, M., A. Bies and S. Kulick. 2008. Enhancing the Arabic Treebank: A Collaborative Effort toward New Annotation Guidelines. *Proceedings of the International Conference on Language Resources and Evaluation*.
- Maamouri, M., A. Bies and T. Buckwalter. 2004. The Penn Arabic Treebank : Building a Large-scale Annotated Arabic Corpus. *Proceedings of the Network for Euro-Mediterranean Language Resources Conference on Arabic Language Resources*. Cairo, Egypt.
- Maamouri, M., A. Bies, T. Buckwalter, M. Diab, N. Habash, O. Rambow and D. Tabessi. 2006. Developing and Using a Pilot Dialectal Arabic Treebank. *Proceedings of the Language Resource and Evaluation Conference*. Genoa, Italy.
- MacWhinney, B. 2001. From CHILDES to Talkbank. *Research on Child Language Acquisition*. M. Almgren, A. Barreña, M. Ezeizaberrena, I. Idiazabal and B. MacWhinney eds., pp. 17-34. Somerville, MA: Cascadilla.
- Maeda, K., H. Lee, J. Medero and S. Strassel, 2006. A new phase in annotation tool development at the Linguistic Data Consortium: The evolution of the Annotation Graph

Toolkit. *Proceedings of the Fifth International Conference on Language Resources and Evaluation*.

Maeda, K., H. Lee, S. Medero, J. Medero, R. Parker and S. Strassel, 2008. Annotation tool development for large-scale corpus creation projects at the Linguistic Data Consortium. *Proceedings of the Sixth International Conference on Language Resources and Evaluation*.

Maeda, K., X. Ma and S. Strassel, 2008a. Creating sentence-aligned parallel text corpora from a large archive of potential parallel text using BITS and Champollion. *Proceedings of the Sixth International Conference on Language Resources and Evaluation*.

Makhoul, J., A. Baron, I. Bulyko, L. Nguyen, L. Ramshaw, D. Stallard, R. Schwartz and B. Xiang. 2005. The effects of speech recognition and punctuation on information extraction performance. *Proceedings of the European Conference on Speech Communication and Technology (Eurospeech)* pp. 57–60.

Marcus, M. G. Kim, M.A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz and B. Schasberger. 1994. The Penn Treebank: annotating predicate argument structure. *Proceedings of the Human Language Technologies Workshop*.

Marcus, M., B. Santorini and M. A. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics* 19: 313-330.

Marcus, M., G. Kim, M.A. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz and B. Schasberger. 1994. The Penn Treebank: Annotating Predicate Argument Structure. *Proceedings of the Human Language Technology Workshop*. San Francisco, California.

Matsoukas, S., I. Bulyko, B. Xiang, K. Nguyen, R. Schwartz and J. Makhoul. 2007. Integrating Speech Recognition and Machine Translation. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing* pp. 1281–1284.

Matsuzaki, T., Y. Miyao and J. Tsujii. 2005. Probabilistic CFG with Latent Annotations. *Proceedings of the Meeting of the Association for Computational Linguistics*.

Matusov, E., A. Mauser and H. Ney. 2006. Automatic Sentence Segmentation And Punctuation Prediction For Spoken Language Translation. *Proceedings of the International Workshop on Spoken Language Translation* pp. 158–165.

Matusov, E., D. Hillard, M. Magimai-Doss, D. Hakkani-Tur, M. Ostendorf and H. Ney. 2007. Improving Speech Translation by Automatic Boundary Prediction. *Proceedings of the International Speech Communication Association Conference (Interspeech)* pp. 2449–2452.

Mauser, A., R. Zens, E. Matusov, S. Hasan and H. Ney. 2006. The RWTH Statistical Machine Translation System for the IWSLT 2006 Evaluation. *Proceedings of the International Workshop on Spoken Language Translation* pp. 103– 110.

- McClosky, D., E. Charniak and M. Johnson. 2006. Effective Self-training for Parsing. *Proceedings of the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics*.
- Melamed, D. 1998. *Annotation Style Guide for the Blinker Project, IRCS Technical Report #98-06*. <http://www.cs.nyu.edu/~melamed/ftp/papers/styleguide.ps.gz>
- Meyers, A., R. Reeves, C. Macleod, R. Szekely, V. Zielinska, B. Young and R. Grishman. 2004. The NomBank Project: An Interim Report. *Proceedings of the Frontiers in Corpus Annotation Workshop, held in conjunction with the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics*
- Mohit, B. and R. Hwa. 2007. Localization of Difficult-to-translate Phrases. *Proceedings of the Second Workshop on Statistical Machine Translation* pp. 248–255, Prague, Czech Republic.
- Mohri, M., F. Pereira and M. Riley. 1998. A Rational Design for a Weighted Finite-state Transducer Library. *Automata Implementation, Lecture Notes in Computer Science* 1436:144– 58. D.Wood and S. Yu, eds. Springer.
- Nichols, C. and R. Hwa. 2005. Word Alignment and Cross-lingual Resource Acquisition. *Proceedings of the Meeting of the Association for Computational Linguistics* Poster and Demonstration Sessions.
- Nigram, K. and R. Ghani. 2000. Analyzing the Effectiveness and Applicability of Co-training. *Proceedings of the Conference on Information and Knowledge Management*.
- Nivre, J., J. Hall, J. Nilsson, A. Chanev, G. Eryigit, S. Kübler, S. Marinov and E. Marsi. 2007. MaltParser: A language-independent System for Data-driven Dependency Parsing. *Natural Language Engineering* 13(2), 95-135.
- Packard, J. 2000. *The Morphology of Chinese*. Cambridge University Press.
- Pajas, P. 2002. Tree Editor TrEd. Charles University, Prague. <http://ufal.mff.cuni.cz/~pajas/tred/>.
- Palmer, M., D. Gildea and P. Kingsbury. 2005. The Proposition Bank: A Corpus Annotated with Semantic Roles. *Computational Linguistics* 31(1).
- Palmer, M., H. Dang and C. Fellbaum. 2007. Making Fine-grained and Coarse-grained Sense Distinctions, Both Manually and Automatically. *Journal of Natural Language Engineering* 13:2, 137-163.
- Palmer, M., J. Hwang, S. W. Brown, K. K. Schuler and A. Lanfranchi, 2009 Leveraging Lexical Resources for the Detection of Event Relations. *Proceedings of the AAAI 2009 Spring Symposium on Learning by Reading*.
- Penman, R.B. 2009. *Web Scraping Made Simple with SiteScraper*. <http://code.google.com/p/site-scraper/>.

Petrov, S., L. Barrett, R. Thibaux and D. Klein. 2006. Learning Accurate, Compact, and Interpretable Tree Annotation. *Proceedings of the Meeting of the Association for Computational Linguistics* pp. 433–440. Sydney, Australia, July.

Petrov, S. and D. Klein. 2007. Discriminative Log-Linear Grammars with Latent Variables. *Proceedings of the Conference on Neural Information Processing Systems*.

Petrov, S. and D. Klein. 2007a. Improved Inference for Unlexicalized Parsing. *Proceedings of the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics*.

Petrov, S. and D. Klein. 2008. Sparse Multi-Scale Grammars for Discriminative Latent Variable Parsing. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*.

Philpot, A., E.H. Hovy and P. Pantel. 2005. The Omega Ontology. *Proceedings of the International Joint Conference on Natural Language Processing ONTOLEX Workshop*.

Pradhan, S., L. Ramshaw, R. Weischedel, J. MacBride, L. Miccuilla. 2007. Unrestricted Coreference: Identifying Entities and Events in OntoNotes. *Proceedings of the International Conference on Semantic Computing*.

Pradhan, S., E. Hovy, M. Marcus, M. Palmer, L. Ramshaw and R. Weischedel. 2007a. OntoNotes: A Unified Relational Representation. *International Journal of Semantic Computing* 1:4 405-419

Pradhan, S., E. Loper, D. Dligach and M. Palmer. 2007b. SemEval-2007 Task 17: English Lexical Sample, SRL and All Words. *Proceedings of the International Workshop on Semantic Evaluations*.

Pradhan, S., W. Ward, K. Hacioglu, J. Martin, D. Jurafsky. 2005. Semantic Role Labeling Using Different Syntactic Views. *Proceedings of Meeting of the Association for Computational Linguistics*.

Prescher, D. 2005. Inducing Head-Driven PCFGs with Latent Heads: Refining a Tree-Bank Grammar for Parsing. *Proceedings of the European Conference on Machine Learning*.

Reeder, F., B. Dorr, D. Farwell, N. Habash, S. Helmreich, E.H. Hovy, L. Levin, T. Mitamura, K. Miller, O. Rambow, A. Siddharthan. 2004. Interlingual Annotation for MT Development. *Proceedings of the Meeting of the Association for Machine Translation of the Americas*.

Roark, B., M. Harper, E. Charniak, B. Dorr, M. Johnson, J.G. Kahn, M. Ostendorf, Y. Liu, J. Hale, A. Krasnyanskaya, M. Lease, I. Shafran, M. Snover, R. Stewart and L. Yung. 2006. SParseval: Evaluation Metrics for Parsing Speech. *Proceedings of the International Conference on Language Resources and Evaluation*. Genoa, Italy.

Roark, B., Y. Liu, M. P. Harper, R. Stewart, M. Lease, M. Snover, I. Shafran, B. Dorr, J. Hale, A. Krasnyanskaya and L. Yung. 2006a. Reranking for Sentence Boundary Detection

- in Conversational Speech. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing* pp. 545–548.
- Rosenberg, A., M. Sharifi and J. Hirschberg. 2007. Varying Input Segmentation for Story Boundary Detection in English, Arabic, and Mandarin Broadcast News. *Proceedings of the International Speech Communication Association Conference (Interspeech)* pp. 2589–2592.
- Roth, R., O. Rambow, N. Habash, M. Diab and C. Rudin. 2008. Arabic Morphological Tagging, Diacritization, and Lemmatization Using Lexeme Models and Feature Ranking. *Proceedings of the Meeting of the Association for Computational Linguistics Short Papers*. Columbus, Ohio.
- Sadat, F. and N. Habash. 2006. Combination of Arabic Preprocessing Schemes for Statistical Machine Translation. *Proceedings of the International Conference on Computational Linguistics and Meeting of the Association for Computational Linguistics* pp. 1–8. Sydney, Australia.
- Sarkar, A. 2001. Applying Co-training Methods to Statistical Parsing. *Proceedings of the Meeting of the North American Chapter of the Association for Computational Linguistics*.
- Shriberg, E., R. Dhillon, S. Bhagat, J. Ang and H. Carvey. 2004. The ICSI meeting recorder dialog act (MRDA) Corpus. *Proceedings of the Special Interest Group on Discourse and Dialogue Conference*.
- Smrž, O. 2007. Functional Arabic Morphology. Formal System and Implementation. *Ph.D. thesis, Charles University in Prague, Prague, Czech Republic*.
- Sproat, R. 1995. Lextools: Tools for finite-state linguistic analysis. *Technical Report 11522-951108- 10TM*, Bell Laboratories.
- Sproat, R. and C. L. Shih. 1990. A Statistical Method for Finding Word Boundaries in Chinese Text. *Computer Processing of Chinese and Oriental Languages* 4(4):336–351.
- Sproat, R. and T. Emerson. 2003. The First International Chinese Word Segmentation Bakeoff. *Proceedings of the Second SIGHAN Workshop on Chinese Language Processing*. Sapporo, Japan.
- Sproat, R., C. Shih, W. Gale and N. Chang. 1996. A Stochastic Finite-State Word-segmentation Algorithm for Chinese. *Computational Linguistics* 22(3):377–404.
- Steedman, M., M. Osborne, A. Sarkar, S. Clark, R. Hwa, J. Hockenmaier, P. Ruhlen, S. Baker and J. Crim. 2003. Bootstrapping statistical parsers from small datasets. *Proceedings of the Meeting of the European Chapter of the Association for Computational Linguistics*.
- Stolcke, A. and E. Shriberg. 1996. Automatic Linguistic Segmentation of Conversational Speech. *Proceedings of the International Conference on Spoken Language Processing* pp. 005–1008.

- Strassel, S. 2004. Linguistic Resources for Effective, Affordable, Reusable Speech-to-Text. *Proceedings of the International Conference on Language Resources and Evaluation*. Lisbon, Portugal.
- Strassel, S. and Cole, A. W. 2006. Corpus Development and Publication. *Proceedings of the International Conference on Language Resources and Evaluation*.
- Strassel, S. *et al.* 2006. Integrated Linguistic Resources for Language Exploitation Technologies. *Proceedings of the International Conference on Language Resources and Evaluation*.
- Stroppa, N. and A. Way. 2006. MATREX: DCU Machine Translation System for IWSLT 2006. *Proceedings of the International Workshop on Spoken Language Translation* pp. 31–36. Kyoto, Japan.
- Tomalin, M. and P. Woodland. 2006. Discriminatively Trained Gaussian Mixture Models for Sentence Boundary Detection. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing* pp. 549–552.
- Tranter, S. and D. Reynolds. 2006. An Overview of Automatic Speaker Diarization Systems. *IEEE Transactions on Audio, Speech, and Language Processing* 14(5):1557–1565.
- Tseng, H., P.C. Chang, G. Andrew, D. Jurafsky and C. Manning. 2005. A Conditional Random Field Word Segmenter. *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.
- Vilar, D., D. Stein, Y. Zhang, E. Matusov, A. Mauser, O. Bender, S. Mansour and H. Ney. 2008. *The RWTH Machine Translation System for IWSLT 2008*. *Proceedings of the International Workshop on Spoken Language Translation* pp. 108–115. Hawaii, USA.
- Wang, W. 2008. Weakly supervised training for parsing Mandarin broadcast transcripts. *Proceedings of the International Speech Communication Association Conference (Interspeech)*. Brisbane, Australia.
- Wang, W., Z. Huang and M. P. Harper. 2007. Semisupervised Learning for Part-of-speech Tagging of Mandarin Transcribed Speech. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*.
- Xia, F. 2000. The Segmentation Guidelines for the Penn Chinese Treebank (3.0). *University of Pennsylvania Institute for Research in Cognitive Science Technical Report* No. IRCS-00-06.
- Xue, N. 2003. Chinese Word Segmentation as Character Tagging. *International Journal of Computational Linguistics and Chinese Language Processing* 8(1):29–48.
- Xue, N. and L. Shen. 2003. Chinese Word Segmentation as LMR Tagging. *Proceedings of the 2nd SIGHAN Workshop on Chinese Language Processing*. Sapporo, Japan.
- Xue, N. and M. Palmer. 2009. Adding Semantic Roles to the Chinese Treebank. *Natural Language Engineering* 15(1): 143-172.

Xue, N., F. Xia, C. Chiou and M. Palmer. 2005. The Penn Chinese Treebank: Phrase Structure Annotation of a Large Corpus. *Natural language Engineering* 11(2): 207-238.

Xue, Ni., F.D. Chiou and M. Palmer. 2002. Building a Large-scale Annotated Chinese Corpus. *Proceedings of the Meeting of the Association for Computational Linguistics*.

Zens, R. and H. Ney. 2006. Discriminative Reordering Models for Statistical Machine Translation. *Proceedings of the Human Language Technologies Conference/ Meeting of the North American Chapter of the Association for Computational Linguistics Workshop on Statistical Machine Translation* pp. 55–63.

Zhang, B. and J.G. Kahn. 2008. Evaluation of Decatur Text Normalizer for Language Model Training. *Technical report, University of Washington*.

Zhao, H. and C. Kit. 2008. Unsupervised Segmentation Helps Supervised Learning of Character Tagging For Word Segmentation and Named Entity Recognition. *Proceedings of the Sixth SIGHAN Workshop on Chinese Language Processing* pp. 106–111. Hyderabad, India.

Zhao, H., C.N. Huang, M. Li and B.L. Lu. 2006. Effective Tag Set Selection in Chinese Word Segmentation via Conditional Random Field Modeling. *Proceedings of the 20th Pacific Asia Conference on Language, Information and Computation* pp. 87–94. Wuhan, China.

Zhou, Q. 2003. Build a Large-scale Syntactically Annotated Chinese Corpus. *Springer Lecture Notes in Computer Science 2807*: 106–113.

Zhu, J. and E.H. Hovy. 2007. Active Learning for Word Sense Disambiguation with Methods for Addressing the Class Imbalance Problem. *Proceedings of the Conference on Empirical Methods on Natural Language Processing*.

Zhu, J., H. Wang and E.H. Hovy. 2008. Multi-Criteria-based Strategy to Stop Active Learning for Data Annotation. *Proceedings of the International Conference on Computational Linguistics*.

Zimmermann, M., D. Tur, J. Fung, N. Mirghafori, L. Gottlieb, E. Shriberg and Y. Liu. 2006. The ICSI+ Multilingual Sentence Segmentation System. *Proceedings of the International Speech Communication Association Conference (Interspeech)*.

Chapter 2: Machine Translation from Text

Editors: Nizar Habash, Joseph Olive, Caitlin Christianson, and John McCary

2.1 Introduction

Author: Nizar Habash

Machine translation (MT) from text, the topic of this chapter, is perhaps the heart of the GALE project. Beyond being a well defined application that stands on its own, MT from text is the link between the automatic speech recognition component and the distillation component. The focus of MT in GALE is on translating from Arabic or Chinese to English. The three languages represent a wide range of linguistic diversity and make the GALE MT task rather challenging and exciting.

There are two big camps when it comes to how to do MT: rule-based MT (RBMT) and statistical machine translation (SMT). RBMT is characterized with the explicit use and manual creation of linguistically informed rules and representations. In its pure form, RBMT includes techniques such as Transfer MT and Interlingual MT. SMT in its pure form is corpus based, i.e., learned from examples of translations called parallel/bilingual corpora. The distinction and nomenclature of both of camps can be deceptive since explicit linguistic rules can be probabilistic and can be learned automatically, of course. There has been a large push for integrating components from the two camps into what is called hybrid approaches. Currently the most successful of such attempts and the general MT research direction in GALE, builds on statistical corpus-based approaches by adding linguistics constraints or features.

The rest of this introduction will contextualize the reader with the basic story of how translation happens in the SMT pipeline. This is important to understand the relevance of the different sections in this chapter, which will be briefly described at the end of this section.

The most important resource in the SMT approach is the corpus of paired source and target texts or parallel corpus (e.g., Arabic and English versions of the UN corpus). An initial step before a parallel text can be used involves cleaning it and preprocessing it to a representation that allow us to learn from it optimally. This includes identifying pairs of sentences within the paragraphs and reducing the sparsity in the text through some form of tokenization. Next comes the automatic alignment step, which learns in an unsupervised manner what pairs of words in the paired sentences are translations of each other. Once this is completed, a translation model can be learned that relates words and sequences

of words in the source language to words or sequences of words in the target language. Translation models typically include various additional statistics reflecting the likelihood of a certain translation pair, which are learned from the corpus. A translation model relates source to target language, but it does not model the fluency of the target language. To address this, a different type of a model, called a language model is used to rank complete sentences in the target language. In their simplest form, language models are built for sequences of words or n-grams. Since language models are monolingual they typically can be built with much more data than parallel text. Translation of a sentence then involves combining entries from the translation and language models to produce target sentences in what is called decoding. The search space for decoding can be very large especially when techniques such as distortion, which allow words to move around to model syntactic differences, are used. Efficient and accurate decoding for MT is an important area of research. The output of a decoder is a ranked list of optimal sentences in the target language. Every single step described here uses numerous parameters that can be manually set or automatically learned to maximize the system's overall performance against some automatic evaluation measure. Linguistically informed constraints, or features, can be used also to help each and every step from preprocessing and alignment through decoding. Finally, since in the GALE project, there are many researchers working on MT in each competing team, an important area in MT research is MT system combination, which investigates how to use the output from multiple systems to produce an output superior to all of them.

This chapter addresses different aspects of the SMT pipeline. 2.2 addresses the question of the optimal token form for Arabic and Chinese to learn MT over. The challenges for Arabic and Chinese are different but the goal is the same: learning better translation models that are robust to the data sparsity challenge. Improvements to the automatic word alignment problem are presented in 2.3. Automatic word alignment is truly the backbone of statistical MT. However, improvements in it are hard to translate to MT improvements. In 2.4, six translation models are presented. These models vary widely in the degree and type of linguistic constraints or back offs they use. 2.5 presents important extensions to simple n-gram language models that address efficiency and linguistic abstraction/depth. 2.6 discusses issues and improvements to the MT decoding problem. 2.7 presents two takes on the issue of how to optimally select training data adapted to the test text. Since SMT systems tend to be biased to the training corpora used to build them, these approaches try to anticipate this bias and use it. Finally 2.8 presents a number of approaches used in different GALE teams to combine the various systems developed to maximize overall performance.

2.2 Segmentation, Tokenization and Preprocessing

2.2.1. Preprocessing Arabic for Arabic-English Statistical Machine Translation

Authors: Adrià de Gispert, William Byrne, Jinxi Xu, Rabih Zbib, John Makhoul, Achraf Chalabi, Hazem Nader, Nizar Habash, Fatiha Sadat

2.2.1.1 Introduction

Statistical machine translation (SMT) is robust when it comes to the choice of input representation: the only requirement is consistency between training and evaluation. This leaves a wide range of possible preprocessing choices, even more so for morphologically rich languages such as Arabic. We use the term ‘preprocessing’ to describe various input modifications that can be applied to raw training and evaluation texts for SMT to make them suitable for model training and decoding, including different kinds of tokenization, stemming, part-of-speech (POS) tagging and lemmatization. We refer to a specific kind of preprocessing as a ‘scheme’ and differentiate it from the ‘technique’ used to obtain it. Identifying a proper preprocessing scheme and technique is important to SMT because tokenized words are the most basic units of translation. Improper tokenization methods can cause a high out-of-vocabulary (OOV) rate and the loss of useful lexical context and hence hurt MT performance.

In this section, we will examine the impact of morphological preprocessing. In our experiments, we used two morphological disambiguation systems, Sakhr Morphological Tagger and MADA (Habash and Rambow 2005). While the use of morphological preprocessing of Arabic text for SMT can be traced at least to 2003 (Lee *et al.* 2003), we hope that our findings using three different techniques on small and medium sized data sets and two different sophisticated analyzers/disambiguators on very large data sets will complement existing work in this area.

2.2.1.2 Linguistic Facts and Motivation

Arabic is a morphologically complex language with a large set of morphological features. These features are realized using both concatenative (affixes and stems) and templatic (root and patterns) morphology with a variety of morphological and phonological adjustments that appear in word orthography and interact with orthographic variations.

We focus here on Arabic's set of attachable function words (or clitics) distinct from inflectional features such as gender, number, person and voice. These clitics are written attached to the word, and thus, increase its ambiguity. We can classify three degrees of cliticization that are applicable in a strict order to a word base:

[CONJ+ [PART+ [Al+ BASE +PRON]]]

At the deepest level, the BASE can have a definite article *Al+* 'the' or a member of the class of pronominal enclitics, +PRON, (e.g. *+hm* 'their/them'. Pronominal enclitics can attach to nouns (as possessives) or verbs and prepositions (as objects). The definite article doesn't apply to verbs or prepositions. +PRON and Al+ do not co-exist on nouns. Next comes the class of particle proclitics (PART+), which includes single-letter prepositions and other functional particles, e.g., *b+* 'by/with' and *s+* 'will'. At the shallowest level of attachment we find the conjunctions (CONJ+) *w+* 'and' *f+* 'so'. They can attach to all words. An archaic question particle proclitic *>+* can attach before conjunctions, but it has almost completely disappeared in modern standard Arabic.

Clitics sometimes interact with the phonological and orthographic forms of the word they attach too. One common example is the feminine ending *+p* (Ta Marbuta), which can only be word final. In medial position, it is turned into the letter *+t* (see example 3 second reading below). Another example is the deletion of the Alif (A) of the definite article *Al+* when preceded by the proclitic *l+*, but no other proclitic (see example 4 first reading below).

The following examples show the decomposition into clitics and base words including cases of orthographic adjustment (bolded) and ambiguous readings that results in different decompositions:

- | | | |
|---------------|--------------------------------|------------------------------|
| 1. wsyqAblhm: | w+ s+ yqAbl +hm | |
| | and will he-meet them | |
| | <i>'and he will meet them'</i> | |
| 2. wbydh: | w+ b+ yd +h | |
| | and with hand his | |
| | <i>'and with his hand'</i> | |
| 3. wktbthA: | w+ ktbt +hA | w+ kt bp +hA |
| | and wrote-she it | and scribes her |
| | <i>'and she wrote it'</i> | <i>'and her scribes'</i> |
| 4. wlljnp: | w+ l+ Al+ jnp | w+ l+ ljnp |
| | and for the paradise | and for a-committee |
| | <i>'and for paradise'</i> | <i>'and for a committee'</i> |

All of these *phrases* are written as one word in Arabic.

These phenomena and examples highlight three issues related to preprocessing. First, ambiguity in Arabic words is an important issue to address. For example, to determine whether a clitic should be split off requires that we determine that said clitic is indeed present in the word we are considering in context – not just that it is possible given an analyzer or, worse, because of regular expression matching.

Secondly, once a specific analysis is determined, the process of splitting off or abstracting off a word feature must be clear on what the form of the resulting word should be so that no unnecessary ambiguity is added.

Third, the degree of splitting to consider (e.g., only conjunctions, conjunctions and prepositions, etc.) needs to be determined empirically and may vary from system to system and depending on the size of available corpora. Under splitting increases sparsity, while over splitting adds ambiguity and requires higher accuracy of preprocessing tools which, when failed, introduce errors and noise.

For a given parallel text, the Arabic vocabulary size is significantly larger than the English one. In the LDC parallel news corpora used in the experiments described below, for instance, the average English sentence length is 33 words compared to 25 words on the Arabic side. The larger vocabulary size on the Arabic side creates problems in sparsity and variability in estimating translation models, since Arabic words do not appear as often in the training data as their English counterparts. This is a general problem when applying statistical machine translation systems to language pair with significant vocabulary size difference. If a language F has a larger vocabulary than a language E , problems in analysis dominate when translating from F into E (due to a relatively larger number of out-of-vocabulary words in the input) whereas, generation problems dominate when translating from E into F .

Several authors have addressed these issues by attempting to reduce the differences in vocabulary size between language pairs. Broadly speaking, the general trend is to find procedures which can perform morphological analysis and disambiguation of the more inflected languages in order to segment words into subword units, with the aim of reducing sparsity in statistical models trained on the data. This potentially reduces the OOV rate, since a compound word might not have occurred in the training data, but its subwords are more likely to occur. These techniques have the largest impact for tasks with relatively small-sized resources although gains are still possible when large parallel corpora are available. For a summary of recent work in this area, see the publications of de Gispert (2006) and de Gispert and Marino (2008).

Note that the application of morphological splitting on the source side is conveniently simpler than applying it on the target side, since no recombination is needed in this case.

2.2.1.3 Arabic Preprocessing: Schemes and Techniques

A *scheme* is a specification of the form of preprocessed output; whereas a *technique* is the method used to create such output. In this section we examine four techniques and nine schemes. Not all schemes can be created using all techniques. Additional schemes we do not discuss here are described by Sadat and Habash (2006).

The different techniques we consider illustrate three degrees of linguistic knowledge dependence. First is a very light and cheap baseline technique. It simply uses greedy regular expression (REGEX) matching to split off prefix/suffix substrings that look like clitics indicated by specific schemes. REGEX technique cannot be used with complex schemes requiring lexical resources.

The second type of techniques is more expensive, requiring the use of a morphological analyzer. The Buckwalter Arabic Morphological Analyzer (BAMA) (Buckwalter 2002) is used to obtain possible word analyses. Using BAMA prevents incorrect greedy REGEX matches. Since BAMA produces multiple analyses, we always select one in a consistent arbitrary manner (first, in a sorted list of analyses).

The third type of techniques is yet more expensive. It involves a disambiguation system that requires an analyzer and a disambiguated training corpus. One such tool is the Morphological Analysis and Disambiguation for Arabic (MADA) tool. MADA is an off-the-shelf resource for Arabic disambiguation (Habash and Rambow 2005), which selects among BAMA analyses using a combination of classifiers trained on the Penn Arabic Treebank (Maamouri and Bies 2004). For BAMA and MADA, applying a preprocessing scheme involves using the general tokenizer tool TOKAN (Habash 2004; Habash 2007a). A second similar tool we experiment with is the Sakhr Morphological Tagger. The Sakhr tagger was trained on a manually tagged corpus of seven million words with morphological, lexical and syntactic features and used maximum likelihood Viterbi decoding to decide the best tag sequence for a sentence.

As for preprocessing schemes, we define next nine preprocessing schemes for Arabic studied in this section. Table 2.1 exemplifies the different schemes for the same sentence.

- ST: Simple Tokenization is the baseline preprocessing scheme. It is limited to splitting off punctuations and numbers from words and removing any diacritics that appear in the input. This scheme requires no disambiguation.
- Decliticizations (D1, D2 and D3): D1 splits off the class of conjunction clitics only. D2 splits off the class of particles beyond D1. Finally, D3 splits off what D2 does in addition to the definite article and pronominal clitics.

- MR: Morphemes. This scheme breaks up words into stem and affixal morphemes. It is identical to the initial tokenization used by Lee (2004).
- EN: English-like. This scheme is intended to minimize differences between Arabic and English. It decliticizes similarly to D3; however, it uses lexeme and English-like POS tags instead of the regenerated word and it indicates the pro-dropped verb subject explicitly as a separate token.
- Sakhr Morphological Analysis is the output of the Sakhr Tagger.
- Split (cliticizing) Morphemes is comparable to D3 except that orthographic adjustments are not done.
- PREF+STEM+SUF is a version of Split Morphemes that collapses the prefixes (specifically proclitics) and suffixes (specifically enclitics).

Arabic Text (ST)	wb*lk ElynA An nEArDhA bkl AlwsA}l
English translation	and thus we have to oppose it with all means
D1	w+ b*lk ElynA An nEArDhA bkl AlwsA}l
D2	w+ b+ *lk ElynA An nEArDhA b+ kl AlwsA}l
D3	w+ b+ *lk EIY +nA An nEArD +hA b+ kl Al+ wsA}l
MR	w+ b+ *lk EIY +nA An n+ EArD +hA b+ kl Al+ wsA}l
EN	w+ b+ *lk/DT EIY/IN +nA An/IN EArD/VBP +S:IP +hA b+ kl/NN Al+ wsylp/NN
Sakhr Morphological Analysis	AND-OR*BE*DEM-SM PREP*PRLG ANN VPC-IDP*PRLN*PRLA & BE*QNTFR AL*N-PF
Split Morphemes	w+ b+ *lk Ely +nA An nEArD +hA b+ kl Al+ wsA}l
PREF+STEM+SUFF	wb+ *lk Ely +nA An nEArD +hA b+ kl Al+ wsA}l

Table 2.1: Example Arabic sentence (in simple tokenization, ST), its translation and different preprocessing schemes: D1, D2 and D3 (different degrees of decliticization), MR (morphemes) and EN (an English-like tokenization), Sakhr morphological processor output and decompositions.

2.2.1.4 Experiments with Preprocessing for Portage

Portage

Portage is a phrase-based SMT system (Sadat *et al.* 2005). For training, Portage uses IBM word alignment models (models 1 and 2) trained in both directions to extract phrase tables. Maximum phrase size used is 8. Trigram language models are implemented using the SRILM toolkit (Stolcke 2002). Decoding weights are optimized using Och's algorithm (Och 2003) to set weights for the four components of the log-linear model: language model, phrase translation model, distortion model and word-length feature. The weights are optimized over the BLEU metric (Papineni *et al.* 2002). The Portage decoder, Canoe, is a dynamic-programming beam search algorithm, resembling the algorithm described by Koehn (2004a).

Experimental Data

For translation model training data, we use an Arabic-English parallel corpus of about five million words available from the Linguistic Data Consortium (LDC): Arabic News (LDC2004T17), eTIRR (LDC2004E72), English translation of Arabic Treebank (LDC2005E46) and Ummah (LDC2004T18). We created the English language model from the English side of the parallel corpus together with 116 million words from the English Gigaword Corpus (LDC2005T12) and 128 million words from the English side of the UN Parallel corpus (LDC2004E13). Arabic preprocessing was varied using the proposed schemes and techniques. English preprocessing comprised down-casing, separating punctuation from words and splitting off “s”. The same preprocessing was used on the English data for all experiments. Only Arabic preprocessing was varied. Decoding weight optimization was done on 200 sentences from the 2003 NIST MT (MT03) evaluation test set. Testing is done with the 2004 NIST MT evaluation test set (MT04). Both MT03 and MT04 have one Arabic source and four English reference translations. We use the evaluation metric BLEU-4 (Papineni *et al.* 2002).

Results and Discussion

We conducted all possible combinations of schemes and techniques discussed in Section 2.2.1.2 with different training corpus sizes: 1%, 10% and 100%. The results of the experiments are summarized.

Across different schemes, EN performs the best under low-resource condition and D2 performs best under large-resource condition. Across techniques and under low-resource conditions, MADA is better than BAMA, which is better than REGEX. Under large-resource conditions, the differences among techniques are not statistically significant, though they are generally consistent across schemes.

MT04	MADA			BAMA			REGEX		
	1	10	100	1	10	100	1	10	100
ST	9.4	22.9	34.6	9.4	22.9	34.6	9.4	22.9	34.6
D1	13.1	26.9	36.1	12.9	26.5	35.6	11.4	25.5	34.8
D2	14.2	27.7	37.1	13.7	27.9	36.2	12.0	25.5	35.8
D3	16.5	28.7	34.3	15.9	28.3	34.2	13.6	26.1	34.0
MR	11.6	27.5	34.4	14.2	27.5	33.4	n/a	n/a	n/a
EN	17.5	28.4	34.5	16.3	27.9	34.0	n/a	n/a	n/a

Table 2.2: Results of the experiments reported in BLEU. The test set presented is MT04 with three techniques (MADA, BAMA and REGEX) and in three training sizes (1%, 10% and 100%). Rows represent different schemes.

An oracle analysis shows that combination of output from all six schemes has a large potential improvement over all of the different systems, suggesting a high degree of complementarity: for MT04 under MADA with 100% training, a 19% improvement in

BLEU score (from 37.1 in D2 to 44.3) can be reached by selecting for each input sentence the output with the highest sentence-level BLEU score. Obviously, this is an optimistic score that only identifies the topline performance of a combination system. Sadat and Habash (2006) demonstrate that a combination of a large number of schemes can give a BLEU score of 38.67 on the MT04 data set (4.2% relative increase, 22% of the difference to the combination top line). Additional experiments and detailed analysis and discussion are presented by Habash and Sadat (2006).

2.2.1.5 MADA-Based Arabic Word Tokenization for TTM

Experimental Data

The two Agile newswire sets contain around 1500 segments of data from the NIST and GALE sets. The two news group Agile sets contain between 1500 and 3000 segments from the NIST and GALE web test sets. The NIST test sets for news groups contain about 2100 segments each taken from the GALE Phase 1 quarter 4 LDC web data. The NIST newswire test sets contain about 2000 sentences each from the NIST MT02_05 Arabic-to-English test sets. The broadcast news set contains about 21k words of Arabic broadcast news recognized by a STT system. The parallel training data used is from GALE Phase 2.

We used tokenization scheme ‘D2’ in our experiments, which entails only the segmentation of some prefixes, specifically conjunctions and prepositions. Further segmentation of Arabic words into more and shorter tokens was not found to improve translation results, which is consistent with results presented in the previous section.

	without MADA	with MADA
num. sents.	6489720	6489720
num. words	139.3 M	156.8 M
vocabulary	612.4 k	421.6 k

Table 2.3: MADA morphosyntactic processing and Arabic corpus statistics

Table 2.3 compares the GALE Phase 2 parallel corpus statistics with and without the MADA based Arabic tokenization. As is apparent, there is a very significant vocabulary reduction of around 30% when applying tokenization using MADA.

Experimental Results

To study whether this strong Arabic vocabulary reduction benefits our Translation Template Model (TTM) system (Blackwood *et al.* 2008a), various translation experiments for different conditions have been carried out. After MADA-based Arabic corpus processing, alignment and translation models are retrained from scratch to build independent systems for translating from analyzed Arabic into English. Table 2.4 shows the TTM scores obtained in the newswire sets for the baseline system and by applying the MADA tokenization in corpus preprocessing. Note that two different language models

are compared in this experiment. While ‘ptext-v2x1’ is trained on both English sides of the Arabic-English and Chinese-English parallel texts+Giga Word II Xinhua+Giga Word II AFP, ‘ptext-v2x2’ is a much larger model, trained on Bitext UN data+Bitext News+Giga Word II Xinhua+Giga Word II AFP. Clearly, the use of MADA produces significant improvements both in BLEU and TER. As shown in Table 2.4, this positive impact is also present for newsgroup and speech sets, where differences of more than 1.0 BLEU are observed.

		Newswire		News Groups		Broadcast News	
Language Model		BLEU	TER	BLEU	TER	BLEU	TER
baseline	Ptext-v2x1	44.87	46.34	13.18	68.38	17.96	64.93
with MADA	Ptext-v2x1	45.67	45.55	14.16	67.48	18.77	64.72
baseline	Ptext-v2x1	46.33	45.22	11.97	69.27		
with MADA	Ptext-v2x1	47.13	44.48	13.24	68.12		

Table 2.4: Baseline TTM results on newswire, news groups and broadcast news data sets with and without MADA

We also report the effect on the Minimum Error Training (MET) applied to our baseline TTM system. Table 2.5 compares the MET scores for the baseline and the MADA-based system. Clearly, the improved performance of baseline TTM due to morphosyntax-based word tokenization still holds when applying MET, yielding improvements between 0.8 and 1.3 BLEU points for both newswire and newsgroup data. TER scores show similar improvements. Additionally, MET BLEU scores do not fluctuate over the newsgroup data, steadily improving across iterations. These results demonstrate that morphological analysis can improve translation performance over a variety of conditions and that it interacts well with other components of the translation system.

		MET iteration	Newswire		News Groups	
			BLEU	TER	BLEU	TER
Baseline	1	47.50	46.03	5.37	128.20	
with MADA	1	48.73	45.24	14.56	76.06	
Baseline	3	47.98	45.32	14.80	70.02	
with MADA	3	48.70	44.21	16.05	69.50	

Table 2.5: Effect of MET with and without MADA. Tuned on newswires and news groups

Finally, we apply integrated lattice rescoring tools to include several additional models, namely a second-order Phrasal Segmentation Model (PSM) (Blackwood 2008) and a Word Alignment Model 1 implemented via MTTK (Machine Translation Tool Kit) (MOD1) (Deng and Byrne 2006; Deng and Byrne 2008). First, Finite State Machine (FST) composition is used to apply a weighted second-order Phrasal Segmentation Model and a Phrase Insertion Penalty and to search for their optimal weights over the tuning set, producing a new set of lattices. Then, a second alignment of these lattices using weighted

Model 1 word-to-word $p(e|f)$ probabilities (from MTTK) is performed with an additional word insertion penalty (Blackwood *et al.* 2008a). Results comparing these lattice rescoring steps applied to the baseline and the MADA-based system are shown in Table 2.6.

Two basic conclusions can be drawn from these tables. On the one hand, the significant differences already observed in baseline TTM scores between the regular system (without MADA) and the system incorporating MADA-based Arabic tokenization still hold after MET and lattice rescoring is applied. In fact, final BLEU improvements are of about 1.3 points (except in news groups test, where they are 0.9) and TER reductions are more than 1 point for newswire and about 2 points for newsgroup data.

		Newswire		News Groups	
		BLEU	TER	BLEU	TER
Baseline	TTM+MET	47.98	45.32	14.80	70.02
with MADA		48.70	44.21	16.05	69.50
Baseline	+PSM	49.65	45.00	15.52	70.01
with MADA		50.52	44.27	16.13	68.52
Baseline	+PSM+MOD1	49.62	44.72	15.83	69.42
with MADA		50.95	43.56	16.69	67.70

Table 2.6: Effect of Phrasal Segmentation Model lattice rescoring (PSM) and MTTK Model 1 lattice rescoring (MOD1) applied to MET lattices with and without MADA.

While Model 1 rescoring did not significantly improve the results after Phrasal Segmentation Model in the baseline system, this changes when applying it on the MADA version. Clearly, there is a positive interaction between vocabulary reduction and word alignment models. MADA training has improved both the alignment models trained on the underlying parallel text and the translation system based on the alignments. Overall, the use of morphosyntax-based Arabic word tokenization is found to be useful in improving the TTM translation system.

2.2.1.6 Experiments using Sakhr's Morphological Tagger with BBN's Hierarchical SMT System

This section describes experiments on the use of the Sakhr Arabic Morphological Tagger to split prefix and suffix morphemes of Arabic words. We show that splitting the affixes of both training and test data using the Sakhr Morphological tagger reduces out of vocabulary rates and improves both TER and BLEU scores.

Based on the POS information from the Sakhr tagger, the morpheme boundaries can then be easily determined by matching the surface forms of the limited list of affixes. When one morpheme is a substring of another, care has to be taken not to match the substring morpheme before the superstring morpheme. If a word has several prefixes or

suffixes, they are concatenated into a single composite prefix or suffix in order to avoid unnecessarily long sentences that complicate training and decoding. The most general form of a decomposed word is, then, PREF+STEM+SUFF.

The experiments were performed using the BBN Hierarchical SMT system (Shen *et al.* 2008). The system was trained on GALE Phase 2 parallel training data. The vocabulary size was reduced from 0.76 million words to 0.42 million words.

For newswire and web data, we tuned on Agile devset and tested on Agile testsets. All tuning is done on the BLEU score.

Test Set	Affix Splitting	TER (%)	BLEU (%)	OOV Rate
newswire	No	43.81	47.10	0.51%
	Yes	42.82	47.87	0.19%
web	No	69.70	16.77	3.75%
	Yes	63.31	18.61	0.97%

Table 2.7: Impact of morphological analysis using Sakhr tagger on BLEU, TER & OOV rates.

Table 2.7 shows the impact of affix splitting on the TER AND BLEU scores. The improvement in BLEU is 0.8 and 1.8 points for newswire and web data respectively. The TER improvement is around 1% on newswire. On the web data, the TER improvement is very large, 6.4%, which correlates to a large reduction in OOV rate shown in the same Table. The large difference in MT scores between the newswire and web data is mostly due to the existence of four reference translations for the newswire data and only one for the web data.

2.2.1.7 Comparison between Sakhr Morphological Tagger and MADA

We performed experiments to compare the two affix splitting methods on identical test sets and obtained mixed results. We provide only a summary here in order to save space. For the TTM system, MADA is better than SAKHR by about 0.5 BLEU point. However, for the BBN Hierarchical system, SAKHR is better than MADA by approximately 0.5 BLEU point. More experiments are needed to determine if the discrepancy is due to fundamental differences in the systems or small differences in data preprocessing.

2.2.1.8 Conclusions

Experiments with two Arabic morphological analysis and disambiguation systems showed that morphologically aware tokenization significantly reduced data fragmentation and improved Arabic-to-English SMT. One area for future work is to combine different morphological splitting algorithms for Arabic MT. For example, simple MBR hypothesis combination schemes can be used to exploit these differences in morphological analyses

yielding improvements of around 1.5 BLEU relative to that of the best individual system (de Gispert 2009)

2.2.2. Customizing Chinese Word Segmentation for Improved Machine Translation

Authors: Pi-Chuan Chang, Michel Galley, Niyu Ge and Christopher D. Manning

2.2.2.1 Introduction

While occasionally experimenting with other tools, for Chinese-English machine translation (MT), the GALE Rosetta consortium has uniformly used an automatic word segmentation of the Chinese input according to the Chinese Treebank word segmentation standard (Xue *et al.* 2005), as produced by the Stanford Chinese segmenter (Tseng *et al.* 2005). This segmenter is a feature-based conditional random field (CRF) model, which had produced competition-winning performance in formal evaluations (Emerson 2005). However, during the GALE Phase 2 evaluation, we became aware that although the CRF segmenter gives a very good F-measure on a segmentation evaluation and generates a very good recall rate on out-of-vocabulary (OOV) words, it did not operate in a way that was most congenial to improving MT performance, at either training or test time. Since the CRF segmenter tried to identify unknown words and compounds, the issue could be that it produced a large vocabulary relative to a lexicon-based segmenter. However, as the experiments below show, the truth turned out to be more complex than this. Based on a series of experiments, we improved the Stanford CRF segmenter with lexicon-based features that improve the consistency of segmentation and yield better MT performance. Also, since the optimal granularity of words for phrase-based MT is unknown, we introduced an extra feature in the CRF segmenter which we can use to adjust average word size and, thus, can further improve MT performance. The lexicon-based features reduced the MT training lexicon by 29.5%, reduced the MT test data OOV rate by 34.1% and led to a 0.38 BLEU point gain on the test data (MT05). When we tuned the CRF model to adjust word size to directly optimize the performance of MT, we found that words slightly shorter than the CTB standard were optimal and that we could improve performance by 0.52 BLEU (on MT05).

2.2.2.2 Experimental Setting

Chinese Word Segmentation

For directly evaluating segmentation performance, we train each segmenter on the SIGHAN Bakeoff 2006 training data (the UPUC data set) and then evaluate on the test data. The training data contains 509K words and the test data has 155K words. The percentage of words in the test data that are unseen in the training data is 8.8%. Detail of the Bakeoff data sets can be seen in the work of Levow (2006). In order to understand how each segmenter performs on OOV words, we will report the segmentation

F-measure, the in-vocabulary (IV) recall rate, as well as, OOV recall rate of each segmenter.

Phrase-Based Chinese-to-English MT

In order to allow easier comparisons, we used the MT system Moses, an open source, state-of-the-art, phrase-based system (Koehn *et al.* 2003). We build phrase translations by first acquiring bidirectional GIZA++ (Och and Ney 2003) alignments and using Moses' grow-diag alignment symmetrization heuristic.¹ We set the maximum phrase length to a large value (10), because some segmenters described later in this section will result in shorter words, therefore it is more comparable if we increase the maximum phrase length. During decoding, we incorporate the standard eight feature functions of Moses as well as the lexicalized reordering model. We tuned the parameters of these features with Minimum Error Rate Training (MERT) (Och 2003) on the NIST MT03 Evaluation data set (919 sentences) and then test MT performance on NIST MT02 and MT05 Evaluation data (878 and 1082 sentences, respectively). We report MT performance with uncased scores using the original BLEU metric (Papineni *et al.* 2001).

The MT training data was subsampled from the GALE Phase 2 training data using a collection of character 5-grams and smaller n -grams drawn from all segmentations of the test data. Since the MT training data is subsampled with character n -grams, it is not biased towards any particular word segmentation. The MT training data contains 1,140,693 sentence pairs; on the Chinese side there are 60,573,223 non-whitespace characters and the English sentences have 40,629,997 words.

Our main source for training our 5-gram language model was the English Gigaword corpus and we also included close to one million English sentences taken from LDC parallel texts: GALE Phase 1 training data (excluding FOUO data), Sinorama, Xinhua News and Hong Kong News. We restricted the Gigaword corpus to a sub-sample of 25 million sentences, because of memory constraints.

2.2.2.3 Understanding Chinese Word Segmentation for Phrase-Based MT

In this section, we experiment with lexicon-based and feature-based segmenters and find that segmentation performance doesn't always correlate positively with MT performance. We introduce a feature-based CRF that uses lexicon features and show that this segmenter combines desirable characteristics from both the feature-based and lexicon-based segmenters.

¹ In our experiments, this heuristic consistently performed better than the default, grow-diag-final.

Lexicon-Based and Feature-Based Segmenters

We compare the performance of a simple maximum matching lexicon-based model (MaxMatch) and a CRF segmenter with only basic features. We trained a CRF model with a set of basic features: character identity features of the current character, previous character and next character and the conjunction of previous and current characters as zero-order templates. We will refer to this segmenter as CRF-basic.

	CRF-basic	MaxMatch	CRF-Lex
F measure	0.877	0.828	0.940
OOV Recall	0.502	0.012	0.729
IV Recall	0.926	0.951	0.970
MT05 (test) BLEU	30.35	30.73	30.95
MT05 (test) METEOR	58.57	59.08	59.71

Table 2.8: The segmentation and MT performance of CRF-basic, MaxMatch and MaxMatch

Table 2.8 shows that the feature-based segmenter CRF-basic outperforms the lexicon-based MaxMatch by 5.9% relative F-measure because CRF-basic has a greater OOV recall rate. We see that a feature-based segmenter like CRF-basic clearly has a stronger ability to recognize unseen words. On MT performance, however, CRF-basic is 0.38 BLEU points worse than MaxMatch on the test set. The results show that better segmentation F-measure does not always lead to a better MT BLEU score, because the segmentation F-measure ordering is CRF-basic > MaxMatch, while, in terms of MT performance, MaxMatch > CRF-basic.

To better understand what factors other than segmentation F-measure can affect MT performance, we introduce another CRF segmenter CRF-Lex that includes lexicon-based features by using external lexicons. More details of CRF-Lex will be described in the next section. In Table 2.9, we list some statistics of each segmenter to explain this phenomenon. First we look at the lexicon size of the MT training and test data. While segmenting the MT data, CRF-basic generates an MT training lexicon size of 583K unique word tokens and MaxMatch has a much smaller lexicon size of 39K. CRF-Lex uses lexicon features in the feature-based framework, and it successfully reduced the lexicon OOV rate. From Table 2.8 and Table 2.9, we see that the hybrid feature-based segmenter CRF-Lex outperforms MaxMatch in MT performance despite its larger lexicon size. This is because CRF-Lex generalizes better on OOV words and more importantly, has better consistency (Chang *et al.* 2008)

Segmenter	CRF-basic	Maxatch	CRF-Lex
# MT Training LexiconSize	583147	39040	411406
# MT Test Lexicon Size	5443	5083	5164
MT Test Lexicon OOV rate	7.40%	0.49%	4.88%

Table 2.9: MT Lexicon Statistics of three segmenters

Making Use of External Lexicons if a Feature-Based Sequence Model

Lexicon-based Features	Linguistic Features
(1.1) $L_{Begin}(C_n); n \in [-2,1]$	(2.1) $C_n; n \in [-2,1]$
(1.2) $L_{Mid}(C_n); n \in [-2,1]$	(2.2) $C_{n-1}C_n; n \in [-1,1]$
(1.3) $L_{End}(C_n); n \in [-2,1]$	(2.3) $C_{n-2}C_n; n \in [1,2]$
(1.4) $L_{End}(C_{-1}) + L_{End}(C_0) + L_{End}(C_1)$	(2.4) $Single(C_n); n \in [-2,1]$
(1.5) $L_{End}(C_{-2}) + L_{End}(C_{-1}) + L_{Begin}(C_0)$ $+ L_{Mid}(C_0)$	(2.5) $UnknownBigram(C_{-1}C_0)$
(1.6) $L_{End}(C_{-2}) + L_{End}(C_{-1}) + L_{Begin}(C_{-1})$ $+ L_{Begin}(C_0) + L_{Mid}(C_0)$	(2.6) $ProductiveAffixes(C_{-1}C_0)$
	(2.7) $Reduplication(C_{-1}C_n), n \in [0,1]$

Table 2.10: Features for CRF-Lex

In this section we describe how we use external lexicons in our feature-based segmenter to improve its consistency. We add lexicon-based features from external lexicons (i.e., not part of the segmentation training data) to improve the consistency of the CRF model. We built a CRF segmenter with all the features listed in Table 2.10 (CRF-Lex). The linguistic features are adopted from Ng and Low (2004) and Tseng *et al.* (2005) and comprise character identity n -grams, morphological and character reduplication features. Our lexicon-based features are adopted from Shi and Wang (2007), where $L_{Begin}(C_0)$, $L_{Mid}(C_0)$ and $L_{End}(C_0)$ represent the maximum length of words found in a lexicon that contain the current character as either the first, middle or last character and we group any length equal or longer than six together. The linguistic features help capturing words that were unseen to the segmenter, while the lexicon-based features constrain the segmenter with external knowledge of what sequences are likely to be words.

The external lexicons we used for the lexicon-based features come from various sources including named entities collected from Wikipedia and the Chinese section of the UN website, named entities collected by Harbin Institute of Technology (Jiang *et al.* 2005), the ADSO dictionary,² EMM News Explorer,³ Online Chinese Tools,⁴ Online Dictionary from Peking University and HowNet.⁵ There are 423,224 distinct entries in all the external lexicons.

The MT lexicon consistency of CRF-Lex in Table 2.9 shows that the MT training lexicon size has been reduced by 29.5% and the MT test data OOV rate is reduced by 34.1%.

² <http://www.adsotrans.com>

³ <http://press.jrc.it/NewsExplorer/home/en/latest.html>

⁴ www.mandarintools.com

⁵ www.keenage.com

2.2.2.4 Optimal Average Token Length for MT

Since the word segmentation standard under consideration (Chinese Treebank (Xue *et al.* 2005)) was neither specifically designed nor optimized for MT, it seems reasonable to investigate whether a different segmentation granularity segmentation is more effective for MT – many of the differences between segmentation standards turn on to what extent compound words are kept together or split apart. In this section, we present a technique for directly optimizing a segmentation property — characters per token average — for translation quality, which yields significant improvements in MT performance.

Our CRF segmenter makes a binary decision on labels: the prediction $y_t = +1$ indicates the t^{th} character of the sequence is preceded by a space and where $y_t = -1$ indicates there is none. In order to calibrate the average word length produced by our CRF segmenter, i.e., to adjust the rate of word boundary predictions ($y_t = +1$), we apply a relatively simple technique (Minkov *et al.* 2006), originally devised for adjusting the precision/recall tradeoff of any sequential classifier. Specifically, the weight vector w and feature vector of a trained linear sequence classifier are augmented at test time to include new class conditional feature functions to bias the classifier towards particular class labels. In our case, since we wish to adjust the frequency of word boundaries, we simply add a single feature that votes for a word boundary and then we can adjust the weight λ_0 of that feature appropriately so as to maximize MT performance. Table 2.11 displays how changes of the bias parameter λ_0 affect segmentation granularity.⁶ Since we were interested in analyzing the range of MT performance between CTB segmentation and segmentation into individual characters, we performed a grid search in the range between $\lambda_0 = 0$ (maximum-likelihood estimate) and $\lambda_0 = 32$ (a value that is large enough to split the data into 1 character words). For each λ_0 value, we ran an entire MT training and testing cycle, i.e., we resegmented the entire training data, ran GIZA++, acquired phrasal translations that abide to this new segmentation, ran MERT and evaluated on segmented data using the same λ_0 values.

λ_0	-1	0	1	2	4	8	2
len	1.64	1.62	1.61	1.59	1.55	1.37	1

Table 2.11: Effect of the bias parameter λ_0 on the average number of character per token on MT data.

⁶ Note that character-per-token averages provided in the table consider each non-Chinese word (e.g., foreign names, num-bers) as one character, since our segmentation post-processing prevents these tokens from being segmented.

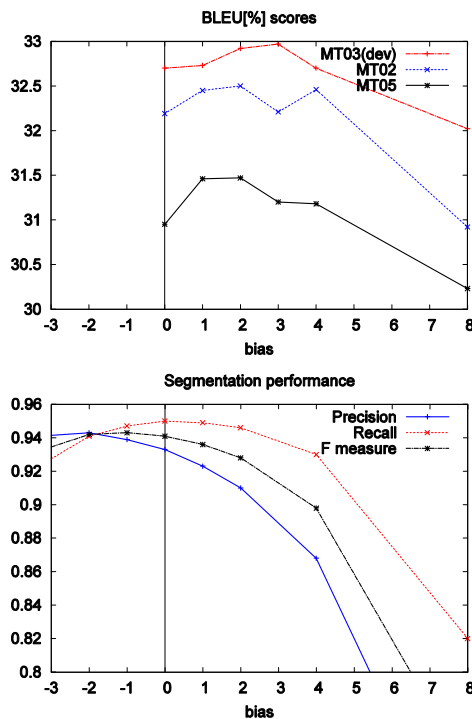


Figure 2.1: A bias towards more segment boundaries ($\lambda_0 > 0$) yields better MT performance and worse segmentation results.

Segmentation and MT results are displayed in Figure 2.1. First, we observe that adjusting the precision and recall tradeoff by setting negative bias values ($\lambda_0 = 2$) slightly improves segmentation performance. We also notice that raising λ_0 yields relatively consistent improvements in MT performance, yet causes segmentation performance (F-measure) to be increasingly worse. While the latter finding is not particularly surprising, it further confirms that segmentation and MT evaluations can yield rather different outcomes. We chose the value $\lambda_0 = 2$ on another dev set (MT02). On the test set MT05, $\lambda_0 = 2$ yields 31.47 BLEU, which represents a quite large improvement compared to the unbiased segmenter (30.95 BLEU). Further reducing the average number of characters per token yields gradual drops of performance until character-level segmentation ($\lambda_0 = 32$, 29.36 BLEU).

Here are some examples of how setting $\lambda_0 = 2$ shortens the words in a way that can help MT.

- separating adjectives and pre-modifying adverbs: 很大(*very big*) → 很(*very*) 大(*big*)
- separating nouns and pre-modifying adjectives: 高血压(*high blood pressure*) → 高(*high*) 血压(*blood pressure*)

- separating compound nouns: 内政部 (*Department of Internal Affairs*) → 内政 (*Internal Affairs*) 部(*Department*).

2.2.2.5 A Comparison across Segmenters

When incorporated into the Rosetta MT systems, the CRF-Lex segmenter shows consistent improvement over other segmenters on the GALE test set DEV07. We finally describe the experimental setup for comparing word segmenters used in Rosetta systems.

Segmenter	#Words	OOV Rate	BLEU _{r1n4-ci}
Character	25,715	0.1%	11.60
Harbin	17,634	2.2%	11.65
FB Maximum Match	17,558	1.2%	11.16
Stanford CRF-Lex	16,768	1.1%	12.10

Table 2.12: Test Data Statistics and Machine Translation Results

Four different segmenters are compared. The first is character segmentation. The second segmenter is similar to a lexicon-based MaxMatch segmenter, but it uses the forward-backward algorithm to maximize the sentence probability of the segmented words. The third segmenter is the Harbin word segmenter by HIT, which uses a mixture of language models and some MAXENT model rescoring (Jiang *et al.* 2005). The fourth segmenter is the Stanford CRF-Lex segmenter. The test data is half of the text portion of GALE DEV07. The test set statistics are shown in Table 2.12. The OOVs in the character input are all non-Chinese words.

The MT decoder used for these experiments is an IBM phrase-based statistical MT system. The first step of the experiment is character-level sub-sampling from the 10M sentence Chinese-English parallel corpus. The sub-sampling selects those parallel sentence pairs that have at least one n -gram overlap with the training data. We match up to 5-gram in the sub-sample process. The sub-sampled training corpus is then segmented by the four segmenters resulting in four sets of training data for each style of segmentation. We then measure the OOV rate of the segmenters on the corresponding training corpus. The HMM model and Block Model (Zhao *et al.* 2005) are used to produce word alignments on the training data. Phrase pair translations are extracted from the word alignments. We use a smoothed 5-gram English LM built on the English Gigaword corpus and the English side the Chinese-English parallel corpora distributed by LDC from years 2000 to 2005. The reordering used in the decoding is phrase level reordering. Up to two phrases are allowed to be skipped over. The metric of evaluation is case insensitive BLEU_{r1n4} with one reference translation. The results are presented in Table 2.12. The numbers are much lower than those for MT03/MT05, because this test set has only one reference translation, whereas the NIST test sets have four reference translations. Here we see that the CRF-Lex segmenter outperforms all other segmenters by a significant margin.

2.2.2.6 Conclusion

We have investigated what segmentation properties can improve machine translation performance. We found that a feature-based CRF segmenter combined with lexicon features has better consistency and the ability to recognize new words. We also found that neither character-based nor a standard word segmentation standard are optimal for MT and show that an intermediate granularity is much more effective. We find that either character segmentation or the state-of-the-art Harbin segmenter performs about 0.5 BLEU_{r1n4} points better than a baseline FB Maximum Match segmenter, but that the improved Stanford segmenter is another 0.5 BLEU_{r1n4} above the performance of these segmenters.

2.2.3. Combination of Alternative Word Segmentations for Chinese Machine Translation

Authors: Jinxi Xu and Roger Bock

2.2.3.1 Introduction

Word segmentation is important to Chinese machine translation (MT) because tokenized words are the most basic units of translation. We address the problem of word segmentation using two techniques. One technique is to segment the parallel training data using an ensemble of complementary segmentation algorithms and combine the alignments in order to mitigate the impact of segmentation errors. The other is to decode input sentences as sequences of characters to reduce mismatches between training examples and test data. While similar ideas appeared in earlier studies (Xu *et al.* 2005a; Zhang *et al.* 2008; Dyer *et al.* 2008), in this work we used segmenters that are complementary to maximize the benefit of combination.

2.2.3.2 Motivation

Unlike in languages written in Western scripts, there are no spaces between words in Chinese text. Word segmentation, i.e., detecting word boundaries in running text, is therefore a prerequisite for Chinese MT. Word segmentation is a tricky issue for a number of reasons. First, due to input ambiguity (e.g., “xiang bei-jing/to Beijing” vs “xiang-bei jing/northward capital”), error-free segmentation is not possible. Second, segmentation inconsistency is a big problem. Due to the often murky distinction between words and phrases, the same string may have multiple valid segmentations. For example, “ai-zhi-bing” can be segmented as either “ai-zhi/AIDS bing/disease” or “ai-zhi-bing/AIDS”. Inconsistent segmentations cause fragmentation of training data and mismatches between training and test. Third, choosing the right word granularity is not easy. While shorter words are desirable with limited training (fewer OOV words), longer

words are more desirable with abundant training (less ambiguity). Determining the right granularity usually requires costly experimentation.

We propose two techniques to address the above issues. One technique is to combine different segmentation algorithms to mitigate the problem of segmentation errors in word alignment. Another technique is to address the issues of segmentation consistency and word granularity by using character-based translation. The two techniques will be described in detail in the next two sections.

2.2.3.3 Combination of Different Segmentation Algorithms

The first technique we use is to combine different segmentation algorithms. Specifically, we tokenize the parallel training data using each segmenter and run GIZA++ separately for each. Then we concatenate the alignment files from different segmenters and use the combined alignment file for training the translation model (i.e., rule extraction). The advantage of this approach is that it mitigates the segmentation errors. While each segmenter may make mistakes for a given string, there is a good chance that one of the segmentations is correct and that the decoder will prefer it over the erroneous ones.

In our experiments, we used the following word segmentation algorithms, which were chosen for their complementary characteristics:

- L2R: Left to right greedy longest word match. This algorithm scans the input left to right for the longest dictionary entry that matches the input and then increments the start position before searching for the next word. It uses a predefined lexicon with 52K Chinese words.
- L2R': The same as L2R, except that it uses a smaller lexicon with 18K words. The lexicon was derived by removing infrequent words (words with a small number of occurrences in the parallel training data) from the L2R lexicon.
- R2L: Similar to L2R, except that it scans words right to left.
- Stat: Uses a trigram language model trained on 5M words of manually segmented text, consisting of the Chinese Treebank corpus and the Academic Sinica Balanced Corpus (Huang and Chen 1995). The algorithm builds a lattice of segmentation alternatives of the input and searches for the segmentation path with the greatest language model score.
- Stat': Similar to Stat, except that it uses a unigram language model trained from the Chinese Gigaword corpus. To train the language model, we first segmented the Gigaword corpus using the L2R method.

Table 2.13 shows the degree of overlap between each pair of segmenters, measured by their F score. If one segmenter outputs a tokens and the other b tokens and the number

of shared tokens is c , their F score is $(\frac{c}{a} + \frac{c}{b})/2$. We can see the segmenters are rather different: the F score varies from 84.3% to 93.8%.

F	L2R	L2R'	R2L	Stat	Stat'
L2R	100	91.5	93.8	85.0	87.9
L2R'	91.5	100	93.0	84.3	86.7
R2L	93.8	93.0	100	87.9	90.8
Stat	85.0	84.3	87.9	100	91.4
Stat'	87.9	86.7	90.8	91.4	100

Table 2.13: Segmentation overlaps (F scores) between different segmenters.

2.2.3.4 Character-Based Translation

The second technique we use is to address the issues of segmentation consistency and word granularity by using character-based translation. Specifically, we convert word-aligned training data into character-aligned data, by splitting Chinese words into characters (English words are not affected) and replicating the alignment links accordingly. We do not run GIZA++ directly on character-tokenized data, because the drastic difference in sentence lengths can result in many unaligned characters. The character-aligned data is then used for model training (i.e., rule extraction). In decoding, the input sentences are translated as sequences of characters. Due to the use of phrases in the translation rules, character-based translation allows the test and training data to be matched at different levels (phrases, words and even subwords). We hypothesize that with more sources of evidence (e.g., English LM score) at its disposal, the decoder can do a better job of selecting the optimal segmentation for a given input than with a fixed word segmentation algorithm.

2.2.3.5 Experiments using Phrasal Decoding

The experiments described in this and the following sections use a number of data sets for tuning and testing. These data sets have been constructed from various GALE and NIST evaluation data sets. They contain different subsets of the Chinese NIST MT04-08 tests, as well as subsets of the GALE Phase 2 development and evaluation data. The Agile Chinese newswire development and test sets are around 1500 sentences each. The Agile Chinese development and test v2 sets are around 2000 sentences each. It is worth noting that when a data set is divided, the division always respects document boundaries, so that no two sentences from the same document end up in different sets.

For fast turnaround, we used a phrase-based decoder in the experiments described in this section. The findings are validated using a slower but more accurate hierarchical decoder in the next section. We used the Agile development set for tuning and the Agile

test set for testing. Translation models were trained using the GALE Phase 2 Chinese parallel training data.

Table 2.14 shows that character-based translation trained with the L2R segmenter is better than word-based translation using the same segmenter. The improvement, as seen by comparing the first and second rows, is 0.3 in BLEU and 0.6 in TER. Combining alignments of different segmenters (Comb for training, Char for decoding in the table) produced an additional improvement of 0.2 in BLEU and 1.1 in TER, as seen by comparing the second and seventh rows. The results show that both segmentation combination and character-based translation improve MT performance. The Comb system is also better than all word-based systems in terms of TER and BLEU.

Training segmenter	Decoding segmenter	BLEU	TER
L2R	L2R	31.2	61.3
L2R	Char	31.5	60.7
L2R'	L2R'	31.3	60.6
R2L	R2L	31.4	60.8
Stat	Stat	30.6	61.6
Stat'	Stat'	31.2	61.0
Comb	Char	31.7	59.6

Table 2.14: Impact of character-based decoding and segmentation combination. In the Comb run, alignments from individual segmenters were combined. Char indicates character-based translation.

2.2.3.6 Experiments with Hierarchical Decoding

To further demonstrate the benefits of character-based translation and segmentation combination, we performed experiments using a state of the art hierarchical decoder (Shen *et al.* 2008). For model training, we used the GALE Phase 3 Chinese parallel data. We used the Agile development v2 set for tuning and the Agile test v2 set for testing. Note that the results in this section are not directly comparable to those in the previous section due to the differences in training and test data.

Training segmenter	Decoding segmenter	BLEU	TER
L2R	L2R	36.7	56.1
L2R	char	37.2	56.0
Comb	char	37.2	55.2

Table 2.15: Impact of character based translation and segmentation combination with a hierarchical decoder. In the Comb run, alignments were trained using three segmenters, L2R, L2R' and Stat.

The results in Table 2.15 show that character-based translation improved over word-based translation by 0.5 in BLEU and 0.1 in TER. By combining alignment files from different segmentations, there is an additional 0.8 improvement in TER, but no improvement

in BLEU. The total improvement over the word-based baseline is 0.5 in BLEU and 0.9 in TER. The improvements are in line with those described in the previous section.

2.2.3.7 Discussion

In our experiments, we concatenated the aligned training data from different segmenters in rule extraction. Obviously, not all segmenters are equally useful. A refinement is to weight the alignments from different segmenters (or exclude some segmenters altogether) based on performances on the development set. While performing such an experiment is straightforward, due to the limitation of time we leave it for future work.

While character-based decoding is similar to decoding a segmentation lattice (Dyer *et al.* 2008), it has two advantages. First, it incurs a smaller overhead in translation table size, since rules with the same unsegmented source string are merged. In our experiments, the character-based phrase translation table (extracted from the combined alignments) was only 40% larger than that of the word-based translation table (left to right word segmentation). Due to a compact translation table, the overhead in decoding speed is also small: Character-based decoding was only 45% slower than the word-based baseline. We expect lattice-based methods will result in a much larger translation table and a significant reduction in decoding speed. A disadvantage with character-based decoding is that it cannot assign weights to different segmentations of the input in scoring translation theories, while lattice-based methods do not have this problem.

2.2.4. Synchronous Learning of Chinese Word Segmentation and Word Alignment

Authors: Jia Xu, Jianfeng Gao, Kristina Toutanova, and Hermann Ney

2.2.4.1 Introduction

Chinese sentences are written in the form of a sequence of Chinese characters; words are not separated by white spaces. This is different from most European languages and poses difficulty in many natural language processing tasks, such as machine translation.

It is difficult to define “correct” Chinese word segmentation (CWS) and various definitions have been proposed. The common solution in Chinese-to-English translation has been to segment the Chinese text using an off-the-shelf CWS method and to apply a standard translation model given the fixed segmentation. The most widely applied method for MT is unigram segmentation, such as segmentation using the LDC (LDC 2003) tool, which requires a manual lexicon containing a list of Chinese words and their frequencies. The lexicon and frequencies are obtained using manually annotated data. This method is sub-optimal for MT, because words out of the manual lexicon cannot be

generated. In addition to unigram segmentation, other methods have been proposed. For example, Gao *et al.* (2005) described an adaptive CWS system and Andrew (2006) and Chang *et al.* (2008) employed a conditional random field model for word segmentation. However, these methods are not specifically developed for the MT application, where Chinese word segmentation and translation model training are separate steps although they influence each other.

In the work of Xu *et al.* (2004), word segmentations are learned from word alignments. We refine this method by integrating the Chinese word segmentation into the word alignment training so that the word segmentation and alignment can be learned synchronously and their effects on each other can be considered in the training. We present a log-linear model derived from a generative model which consists of a word model and two alignment models, representing the monolingual and bilingual information, respectively. The model is trained using Gibbs sampling. Alternative segmentation boundaries and realignments of words due to the change of these boundaries are taken into account in the sampling process. New Chinese words are generated using Dirichlet Process and the lexicon is updated dynamically. In this way, two problems are solved: adaptation to the parallel training corpus and out of vocabulary words.

Our experiments on both large (GALE) and small (IWSLT) data tracks of Chinese-to-English translation show that our method improves the performance of state-of-the-art machine translation systems.

2.2.4.2 Review of the Baseline System

In statistical machine translation, we are given a Chinese sentence in characters $c_1^K = c_1 \dots c_K$ which is to be translated into an English sentence $e_1^I = e_1 \dots e_I$. In order to obtain a more adequate mapping between Chinese and English words, c_1^K is usually segmented into words $f_1^J = f_1 \dots f_J$ in preprocessing.

In our baseline system, we apply the commonly used unigram model to generate the segmentation. Given a manually compiled lexicon containing words and their relative frequencies, the best segmentation is the one that maximizes the joint probability of all words in the sentence, under the assumption that words are independent of each other. However, a human collected lexicon can hardly cover all Chinese words in various domains. Words out of the lexicon list are dropped during word segmentation and might not be able to contribute to the translation any more. Inaccurate word distributions can also result in sub-optimal segmentation.

Once we have segmented the Chinese sentences into words, we train standard alignment models in both directions with GIZA++ (Och and Ney 2002) using models of IBM-1 (Brown *et al.* 1993), HMM (Vogel *et al.* 1996) and IBM-4 (Brown *et al.* 1993). The translation system uses a phrase-based decoder with a log-linear model described by Zens and Ney (2004). The feature weights are tuned on the development

set using a downhill simplex algorithm (Press *et al.* 2002). The language model is a statistical n -gram model estimated using modified Kneser-Ney smoothing.

2.2.4.3 Semi-supervised Word Segmentation

We introduce a semi-supervised approach to perform Chinese word segmentation as illustrated in Figure 2.2. The inputs to the system are the bilingual training data, including a set of Chinese sentences in characters and its English translations, a manual Chinese word lexicon, such as LDC lexicon, as well as the test corpus on character level. First, we segment the Chinese training corpus with a unigram segmenter using the manual lexicon and get an initialized training corpus in words. Then, we perform the synchronous training of Chinese word segmentations and word alignments to maximize the likelihood of a log-linear model. Optimal word segmentations and alignments are generated as outputs. By counting the Chinese word frequencies of the generated training corpus, we obtain a lexicon. To combine this lexicon with a manual lexicon, we interpolate the probabilities of each word entry in both lexicons linearly. This combined lexicon is applied to segment the test corpus using unigram segmentation. The optimal Chinese word segmentations of the training and test data, as well as, the alignments of the training data, are the system outputs, which will be used further into the decoder.

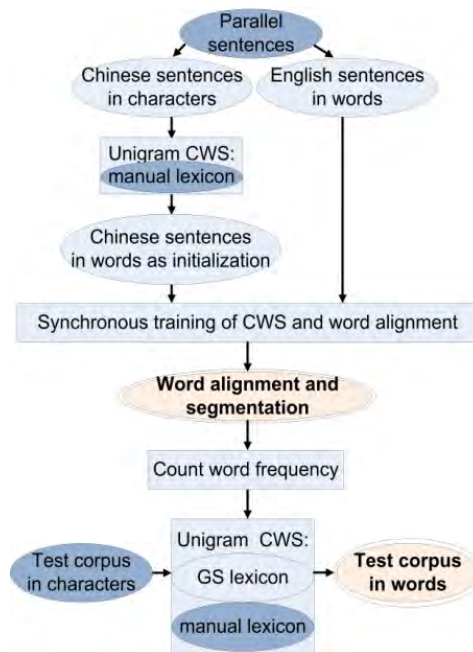


Figure 2.2: Semi-supervised CWS process.

2.2.4.4 Generative and Log-linear Model

Observations		
Chinese characters	c_1^K	小孩玩纸牌
English sentence	e_1^I	Children play cards
Hidden variables		
Alignment normal	a_1^J	e.g. (cards, 纸),(cards, 牌)
Alignment inverse	b_1^I	e.g. (纸, cards),(牌, cards)
Chinese words	f_1^J	e.g. 小孩玩纸牌

Table 2.16: Observations and hidden variables of the generative model.

As a solution to the problems with the conventional approach to CWS mentioned in Section 2.2.4, we propose a generative model for CWS in this section and then extend the model to a more general, but deficient model, a log-linear model in which most features are derived from the sub-models of the generative model.

As shown in Table 2.16, the generative model assumes that a corpus of parallel sentences (c_1^K, e_1^I) is generated along with a hidden sequence of Chinese words f_1^J and a hidden word alignment b_1^I in the inverse direction for every sentence. The joint probability of the observations (c_1^K, e_1^I) can be obtained by summing over all possible values of the hidden variables f_1^J and b_1^I and each value is computed as following:

$$Pr(c_1^K, e_1^I, f_1^J, b_1^I) = Pr(f_1^J) \delta(f_1^J, c_1^K) Pr(e_1^I, b_1^I | f_1^J) \quad (2.1)$$

$$\approx \frac{1}{Z} P(f_1^J)^{\lambda_1} P(e_1^I, b_1^I | f_1^J)^{\lambda_2} P(f_1^J, a_1^J | e_1^I)^{\lambda_3} \quad (2.2)$$

$\delta(f_1^J, c_1^K)$ is 1 if the characters of the sequence of words f_1^J are c_1^K and to 0 otherwise, Z is the normalization factor. We can drop the conditioning on c_1^K in $Pr(e_1^I, b_1^I | f_1^J)$, because the characters are deterministic given the words.

In Equation 2.2, we put the monolingual model and the translation models in both directions together into a single model, where each of the component models is weighted by a scaling factor. This model can be viewed as a weighted linear combination of the log probabilities of sub-models. The weights, which are optimized on a development dataset, have empirical justifications. Since different sub-models are trained on different datasets, their dynamic value ranges can be so different that it is inappropriate to combine their log probabilities through simple addition. Moreover, some models may be poorly estimated due to for example the lack of large amount of training data. Therefore, empirical results

have demonstrated that the use of scaling factors that reflect the relative contribution of different sub-models often improves the performance.

In practice, we do not renormalize probabilities and our model is thus deficient because it does not sum to 1 over observations. However, the model works very well in our experiments. Similar deficient models have been used successfully before, such as in IBM models (Och and Ney 2002).

Monolingual Chinese sentence model

We use the Dirichlet Process unigram word model (Xu *et al.* 2008) to introduce new Chinese word types and to learn word distributions automatically from unlabeled data, where the parameters of a distribution over words G are first drawn from the Dirichlet prior $DP(\alpha, P_0)$. Words are then independently generated according to G . The probability of a sequence of Chinese words in a sentence is thus:

$$Pr(f_1^J) = \prod_{j=1}^J P_G(f_j) \quad (2.3)$$

Translation model

We employ the Dirichlet Process inverse IBM model 1 to generate English words and alignments given the Chinese words. In this model, for every Chinese word f (including the null word), a distribution over English words G_f is first drawn from a Dirichlet Process prior $DP(\alpha, P_0(e))$, where for $P_0(e)$ we use the empirical distribution over English words in the parallel data. Then the probability of an English sentence and alignment given a Chinese sentence in words is given by

$$Pr(e_1^I, b_1^I | f_1^J) = \prod_{i=1}^I \frac{1}{J+1} P_{G_{f_{b_i}}} (e_i | f_{b_i}) \quad (2.4)$$

where the probability of e_i is distributed according to $G_{f_{b_i}}$. This is the same model form as inverse IBM model 1, except we have placed Dirichlet Process priors on the Chinese word specific distributions over English words.⁷

In practice, we observed that using a word alignment model in one direction is not sufficient then added a factor to our model which includes the word alignment in the other direction, i.e., a Dirichlet Process IBM model 1. We ignore the detailed description here, because the calculation is the same as that of the inverse IBM model 1. According to this model, for every English word e (including the null word), a distribution over Chinese words G_e is first drawn from a Dirichlet Process prior $DP(\alpha, P_0(f))$. Here, for

⁷ f_{b_i} is the Chinese word aligned to e_i and $G_{f_{b_i}}$ is the distribution over English words conditioned on the word f_{b_i} . Similarly, e_{a_j} is the English word aligned to f_j in the other direction and $G_{e_{a_j}}$ is the distribution over Chinese words conditioned on e_{a_j} .

the base distribution $P_0(f)$ we used the same spelling model as for the monolingual unigram Dirichlet Process prior as described by Xu *et al.* (2008). The probability of a sequence of Chinese words f_1^J and a word alignment a_1^J given a sequence of English words e_1^J is then computed in the same way.

Gibbs Sampling Training

It is generally impossible to find the most likely segmentation according to our Bayesian model using exact inference, because the hidden variables do not allow exact computation of the integrals. Nonetheless, it is possible to define algorithms using Markov chain Monte Carlo (MCMC) that produce a stream of samples from the posterior distribution of the hidden variables given the observations. We applied the Gibbs sampler (Geman and Geman 1984), one of the simplest MCMC methods, in which transitions between states of the Markov chain result from sampling each component of the state conditioned on the current value of all other variables. For a complete discussion of Gibbs sampling training and the word segmentation and realignment algorithm used in our experiment see (Xu *et al.* 2008).

The Gibbs sampler for our model works as follows: For each iteration we sample on each character position by fixing other segmentations and alignments, then we compare hypotheses considering the segmentation and the related alignments of this position. Each position has two alternative segmentations: a boundary exists, or not. The change of a segmentation boundary causes relinking alignment points to parts or groups of the original words. In the work of Xu *et al.* (2008) all alignment alternatives are discussed in detail. Together with the boundary versus no-boundary state at each character position, we sample a set of alignment links between English words and any of the Chinese words related to this position given all other word alignments and segmentations in the parallel corpus fixed. After sampling by using the posterior probabilities of each candidate, we choose one of these candidates and perform the same operation for the next position. This process is usually terminated until the result is converged. Since we only implemented the IBM model 1 in both directions for computational efficiency, more advanced word alignment models are applied by repeatedly aligning the corpus using GIZA++.

2.2.4.5 Translation Experiments

We performed experiments using our models on a large and a small data track. We evaluated performance by measuring WER (word error rate), PER (position-independent word error rate), BLEU (Papineni *et al.* 2002) and TER (translation error rate) (Snover *et al.* 2006) using multiple references.

Translation Task: Large Track GALE Translation

We first report the experiments on the GALE machine translation task (GALE 2008). The bilingual training corpus is a superset of corpora in the news, conversation domains collected from different sources provided under the GALE program. As shown in Table 2.17, the training corpus in each language contains more than seven million sentences after the bilingual sentence segmentation (Xu *et al.* 2005b). We took LDC (LDC 2003) as baseline method to compare. The word segmentation using Gibbs Sampling (GS) and baseline method generated 92.8 and 93.9 million Chinese running words respectively.

		Chinese		English
		LDC	GS	
Train	Sentences[M]	7.57		
	Running Words[M]	93.9	92.8	102
	Vocabulary[K]	112	121	347
	Singletons[K]	38.1	38.3	152
Test	Sentences	1943		
	Running Words[K]	44.3	44.3	53.2
	Vocabulary[K]	6.78	6.60	6.15
	OOVs (R. W.)	15	17	246
	OOVs (in voc.)	13	14	158

Table 2.17: Statistics of corpora in task GALE.

The CWS model parameters are not optimized but fixed as applied in the IWSLT task because of the computational complexity. The log-linear model scaling factors in the decoder as mentioned in Section 2.2.4.2 are neither optimized and we took the values optimized on the baseline system for convenience. The resulting systems were evaluated on the test corpus in 2008 including all domains with 1943 sentences. We only list the statistics of the first English reference.

Starting from the unigram segmentation as initial word segmentation, we performed Gibbs sampling with only one iteration, which takes several hours, on the training corpus because of the large computational requirement. After that, we merged the GS generated lexicon with a weight of 0.4 and the manual LDC lexicon with a weight of 0.6 using linear combination on the probabilities of the word entries. Then we performed the unigram segmentation on the test corpus using the combined lexicon.

As shown in Table 2.18, on the test data, the BLEU score was improved by 0.5% absolutely or more than 1.8% relatively using GS with combined lexicon. The TER score was also enhanced significantly, 0.8% absolutely and 0.9% relatively.

Method	WER	PER	BLEU	TER
LDC	73.0	49.5	28.2	67.1
Unigram	73.0	49.7	28.4	67.2
GS with combined lexicon	72.5	48.6	28.7	66.3

Table 2.18: Translation performance with the baselines (LDC, unigram) and GS method on GALE [%].

We can see that although the semi-supervised word segmentation is not yet converged, it can still outperform a supervised one in MT. One of the reasons is probably the training and test corpora contain many words and words have different frequencies in our MT data from they do in the manually labeled CWS data.

Task: Small Track IWSLT

The Chinese training corpus of the IWSLT task was segmented using the unigram segmenter as baseline method (Baseline) and our GS method. The parameter optimizations were performed on the Dev2 data with 500 sentences and evaluations were done on both the Dev3 and the Eval data, i.e., the evaluation corpus of (IWSLT 2007).

The model weights of GS were optimized using the Powell (Press *et al.* 2002) algorithm with respect to the BLEU score. We obtained the optimal number of iterations of the GIZA++ word alignment update as four.

Test	Method	WER	PER	BLEU	TER
Dev2	Unigram (Baseline)	38.2	31.2	55.4	37.0
	GS	36.8	30.0	56.6	35.5
Dev3	Unigram (Baseline)	33.5	27.5	60.4	32.1
	GS	32.3	26.6	61.0	31.4
Eval	Characters	49.3	41.8	35.4	47.5
	LDC	46.2	40.0	39.2	45.0
	ICT	45.9	40.4	40.1	44.9
	Unigram (Baseline)	46.8	40.2	41.6	45.6
	9-gram	46.9	40.4	40.1	45.4
	GS	45.9	40.0	41.6	44.8

Table 2.19: Translation performance with different CWS methods on IWSLT[%].

For a fair comparison, we evaluated on various CWS methods including translation on characters, LDC (LDC 2003), ICT (Zhang *et al.* 2003), unigram, 9-gram and GS. Improvements using GS can be seen in Table 2.19. Under all test sets and evaluation criteria, GS outperforms the baseline method. The absolute WER decreases with 1.2% on the Dev3 and with 0.9% on the Eval data over the baseline. In the BLEU score there is no change on the Eval set between the baseline and the GS, because the Eval data has a lower vocabulary coverage with the Dev2 than the other test sets such as the Dev3 do. The optimization of many parameters leads to a slight over-fitting of the model, so that the parameters may not be optimal for the Eval translation.

We compared the translation outputs using the GS with those using the baseline method. On the Eval data, 196 sentences have different translations out of 489 lines, where 64 sentences from the GS are better, 33 sentences are worse and the rests have similar translation qualities. Table 2.20 shows two examples from the Eval corpus. We list segmentations produced by the baseline and the GS methods, as well as the

translations generated using these segmentations. The GS method generates better translation results than the baseline method in these cases.

a) Baseline	有近路吗? do you have a ?
GS	有近路吗? do you have a shorter way ?
REF	is there a shorter route ?
b) Baseline	请告诉我总金额 please show me the in .
GS	请告诉我总金额 please show me the total price .
REF	can you tell me the total amount?

Table 2.20: Segmentation and translation outputs with baseline and GS methods.

2.2.4.6 Conclusion and Future Work

We showed that it is possible to learn Chinese word boundaries during the word alignment training so that the translation performance of Chinese-English MT systems is improved.

We presented a Bayesian generative model for parallel Chinese-English sentences, which uses word segmentation and alignment as hidden variables and incorporates both monolingual and bilingual information to derive word segmentation and alignment for MT.

Starting with initial word segmentation, our method learns both new Chinese words and word distributions using the Dirichlet Process. In a small data environment and a large data environment, our method outperformed the standard Chinese word segmentation approach in terms of the Chinese-English translation quality. In future work, we plan to enrich our models to better represent the true distribution of the data.

2.3 Word Alignment

2.3.1. Word Alignment Revisited

Authors: Francisco Guzman, Qin Gao, Jan Niehues, and Stephan Vogel

2.3.1.1 Introduction

Word alignment can be considered the backbone of Statistical Machine Translation. Even when Statistical Machine Translation (SMT) shifted from a word-based to a phrase-based paradigm, word alignment remained the base for most phrase-based (Koehn *et al.* 2003), hierarchical (Chiang 2007) and syntactic SMT systems (Zollmann and Venugopal 2006; Marcu *et al.* 2006). Generative models have the advantage that they are well suited for a noisy-channel approach. Unsupervised training can be used to align a large amount

of unlabeled parallel corpora. Nonetheless, they have a major disadvantage, because these models are completely unsupervised, they can hardly make use of the increasingly available manual alignments. Also, given their complexity, to incorporate other information, such as POS tags, word frequencies etc., is a non-trivial task. Moreover, because the IBM models are not symmetric, the alignments for different directions are quite different, which makes the search for a symmetrized combination of the word alignments a challenging procedure.

Given the fact that the word alignments serve as a starting point of the SMT pipeline, improving their quality has been a major focus of research in the SMT community. However, the quality of word alignment is not directly related to the quality of translation. In fact, only weak correlation between alignment error rate (AER) and BLEU scores has been reported (Fraser and Marcu 2006b). The mismatch between the quality of word alignment models and that of phrase-based or syntactic based SMT may lead to the phenomenon of “improved” word alignment resulting in decreased translation quality (Vilar *et al.* 2006). This calls for more careful analysis of word alignment errors to better understand the impact of alignment errors and alignment characteristics on the entire MT pipeline. Recently, different efforts have focused on the symmetrization of the word alignment models (Matusov *et al.* 2004; Liang *et al.* 2006b), the inclusion of annotated data in the training of generative models (Fraser and Marcu 2006a) and the use of discriminative models (Blunsom and Cohn 2006; Taskar *et al.* 2005; Niehues and Vogel, 2008). One of the advantages of these discriminative models is that the word alignment quality can be tuned towards a given word alignment quality measurement.⁸ Moreover, their conditional probability model allows the inclusion of different features, enabling that any available knowledge source can be used to find the best alignment.

In this work, we present the results of an extensive error analysis of the word alignments created by the generative models, using the well-known toolkit GIZA++ (Al-Onaizan *et al.* 1999; Gao and Vogel 2008). By characterizing the errors, we hope to shed light on the behavior of the aligners, as well as, to identify some opportunities for improvement. We also present our work on a discriminative word alignment framework, as presented by Niehues and Vogel (2008), which is easy to enhance with new features. We believe that proper analysis of the alignment behavior, coupled with the use of discriminative word alignment, can help to overcome many of the weaknesses of the generative models.

2.3.1.2 Error Analysis

To analyze the errors in automatically generated word alignments we compare the Viterbi alignments generated by GIZA++ against a gold standard of hand aligned data. In

⁸ For some measurements, smoothing is required.

some gold standards, there is a distinction between *sure* and *possible* links. *Sure* links represent the hand alignments made by the annotator for which he is sure of the alignment. *Possible* links are those which represent a degree of uncertainty, e.g., where different annotators differ in the manual alignment. However, in this study, we include only *sure* links. Based on the differences between Viterbi and hand alignments, there are three basic quantities that we can measure: the number of links in which these two alignments agree, i.e., true positives (tp); the number of links that are present in the output of the aligner, but not in the gold standard, i.e., false positives (fp); and the number of links that are present in the gold standard, but not in the output of the aligner, i.e., false negatives (fn). There are several metrics that are used for measuring the quality of a word alignment, but most of them are based on these three quantities. Error measures included Alignment Error Rates, Precision and Recall:

- Alignment Error Rate: AER, as defined by Och and Ney (2003), takes Sure and Possible links into account.

$$AER = \frac{|AUS| + |AUP|}{A+S} \quad (2.5)$$

Without the distinction into sure and possible links, the metric is related to the F measure:

$$AER = 1 - \frac{2tp}{2tp+fp+fn} = 1 - F \quad (2.6)$$

- Precision: This measure gives us a notion of how accurate is the output of the aligner. It is the ratio of the correct to all generated links.

$$Precision = \frac{tp}{tp+fp} \quad (2.7)$$

- Recall: This measures how well we cover the desired links, i.e. those in the hand alignments, with the automatically generated ones. That is, of all the links in the gold standard, what is the amount of links that are also present in the aligner output.

$$Recall = \frac{tp}{tp+fn} \quad (2.8)$$

For some time, AER has been the predominant metric in the area. However, recent studies have raised some concerns. Fraser and Marcu (2006b) showed that when the distinction between sure and possible alignments is made, AER is biased towards precision, rather than being balanced between precision and recall and suggest using weighted F-measure to have control over this balance. In this study, we chose to use AER, given that are not making use of possible alignments and that the optimal coefficient for the weighted F-measure depends on the language being analyzed.

Data Analysis

In this section, we present a detailed error analysis of the Viterbi alignments, resulting from training the standard sequence of word alignment models IBM1, HMM, IBM3 and finally IBM4, in both directions, i.e., source to target (S2T) and target to source (T2S). We used the modified GIZA toolkit (Gao and Vogel, 2008). In addition, we generated the combined alignment, using the grow-diag-final heuristics implemented and used in the MOSES package (Koehn *et al.* 2007). In our analysis we use alignments for Arabic-English and Chinese-English. The data sets used for training and the data sets that we used for evaluation are summarized in Table 2.21.

		Training Corpora		Evaluation Corpora		
		# sentences	# words	# sentences	# words	Avg. Sen. Len. (std)
A to	Arabic	7.7M	218M	14K	287K	20.5 (10.)
E	English	7.7M	216M	14K	358K	25.7 (12.)
C to	Chinese	11.0M	309M	19K	375K	19.4 (12.)
E	English	11.0M	273M	19K	463K	23.9 (15.)

Table 2.21: Statistics of corpora used in GIZA++ training and evaluation test sets

From the statistics we can see that the English sentences are on average longer than the Arabic and Chinese sentences. Therefore, it is to be expected that source words often need to be aligned to several English words (i.e., English words have higher fertility), or that many English words are not aligned to Chinese or Arabic source words. This poses a problem for the generative models, because they can align at most one source word to a target word. Therefore, a greater asymmetry is to be expected.

Alignment	#Links GS	#Links	Precision	Recall	AER
Arabic-English					
IBM4 S2T	336,995	275,386	73.6	60.2	33.7
IBM4 T2S		339,281	68.6	69.0	31.1
combined		334,469	73.2	72.6	27.0
Chinese-English					
IBM4 S2T	527803	359,485	51.9	35.3	57.9
IBM4 T2S		451,222	66.4	56.7	38.7
combined		437,241	67.7	56.1	38.5

Table 2.22: Statistics of the different alignments (GS = Gold Standard Alignment).

In Table 2.22, we summarize the results for the evaluation of Arabic-English and Chinese-English. As expected, IBM4 models from source-to-target perform worse than the target-to-source ones. This difference is more striking for the Chinese-English case, where the AER difference is almost 20 points. What can be observed for the source-to-target cases is that the aligner is omitting many alignments, which leads to low recall,

while precision remains higher. In the hand-aligned data Chinese and Arabic words are often aligned to more than one English word, which cannot be replicated in the asymmetric IBM Models.

Unaligned Words

As an additional consequence of the symmetry we also expect an unbalance between the number of “NULL” alignments, i.e., source words aligned to the virtual “NULL words” and the number of target words that are left unaligned. Looking at the statistics of unaligned words for the output of GIZA++, as well as, the behavior of symmetrization heuristics, we can get a better sense of the impact of the underlying structure of these models. Table 2.23 gives the percentage of NULL-Alignments (words aligned to the NULL word) and the percentage of words not aligned. We also show these numbers for a symmetrized alignment and for the discriminative word alignment (DWA) described later in Section 2.3.1.3.

Alignment	NULL Alignments	Unaligned Words
Arabic-English		
Manual Alignment	8.5%	11.8%
IBM4 S2T	3.5%	30.0%
IBM4 T2S	5.3%	15.7%
combined	5.5%	7.80%
DWA	17.6%	18.7%
Chinese-English		
Manual Alignment	7.80%	11.9%
IBM4 S2T	5.5%	23.8%
IBM4 T2S	6.4%	34.5%
combined	14.6%	14.6%
DWA	16.7%	23.7%

Table 2.23: Statistics for unaligned words and null-alignments

Compared to the hand aligned data, GIZA++ tends to have less NULL alignments, but a larger number of unaligned words. Depending on the language considered as source and target, we see that up to one third of the target words are not aligned. This highlights a weakness of the generative alignment models. On the other hand, the heuristic symmetrization (grow-diag-and) generates alignments which are more balanced between the number of NULL alignments and unaligned words and reasonably close to the hand alignment. In contrast, the discriminative alignment is sparser, leaving more source and target words unaligned. This is especially true for Chinese-English where we observe a large number of unaligned English words, which will impact the phrase pairs extracted from this kind of alignment.

Alignment Errors: Sentence Level Analysis

Now, we analyze the distribution of alignment errors over all sentence pairs. The distribution of the per-sentence AER shown in Figure 2.3 is almost Gaussian. However, we see a peak at AER=0. This is partially an artifact of the bucketing: to achieve an AER of 0.1 the sentence has to be already in the range of 10 words.

Figure 2.4 shows some Arabic-English sentence pairs, which were perfectly aligned. Manual inspection revealed that all the perfectly aligned sentence pairs have no long distance reordering.

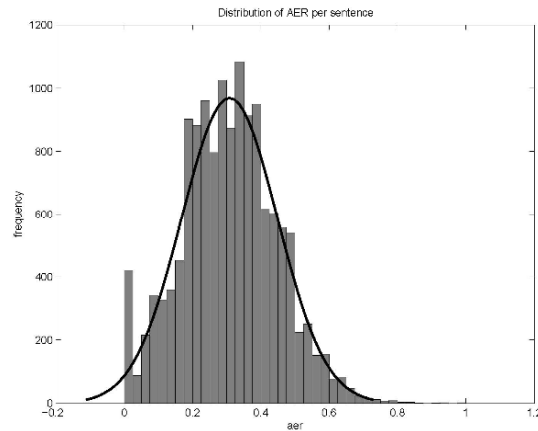


Figure 2.3: AER distribution for the Arabic-English alignment generated by GIZA++ S2T. The vertical axis shows the counts of sentences versus their corresponding AER evaluation in the horizontal axis.

الملك حسين يبدأ المرحلة الرابعة من العلاج الكيميائي	King Hussein Begins Fourth Phase of Chemotherapy
1-1 2-2 3-3 5-4 4-5 6-6 7-7 8-7	
روبن كوك يستبعد أي تدخل للحكومة البريطانية في قضية بينوشيه	Robin Cook Dismisses Any British Government Intervention in Pinochet Case
1-1 2-2 3-3 4-4 7-5 6-6 5-7 8-8 10-9 9-10	
وتستخدم اليونيسكوم في العراق نحو 120 شخصا بينهم 40 مفتشا .	UNSCOM employs about 120 persons in Iraq , including 40 inspectors .
2-1 1-2 5-3 6-4 7-5 3-6 4-7 8-9 9-10 10-11 11-12	

Figure 2.4: Examples of perfectly aligned sentences of the Arabic-English alignment generated by GIZA++ S2T.

Finally, it is interesting to see how well the alignment model scores correlate to the alignment quality. A high correlation would allow us to select sentence pairs, for which

the alignment is more likely correct. In Figure 2.5 a scatter plot shows the correlation between the alignment error rate and the normalized alignment log probability (i.e. divided by the number of words in the sentence). We see a weak correlation, which might not be sufficient for a reliable data selection. We also observe a series of bands (at approx 0.3, 0.5, 0.6, etc.) which correspond to the discrete nature of AER, i.e. a sentence pair of length $2/2$ can only have AERs of f_0 , 0.25, 0.5, 0.75, 1g.

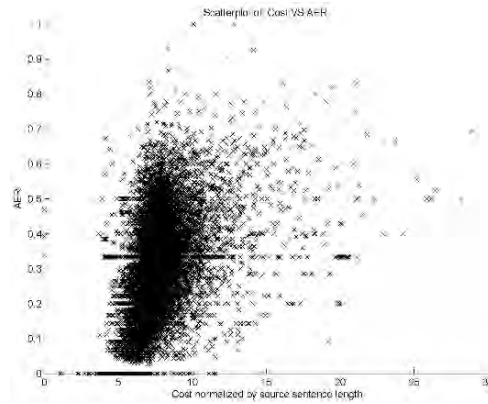


Figure 2.5: Scatter plot of AER per Sentence versus Model Cost of the Arabic-English alignment generated by GIZA++ S2T. In the vertical axis, we observe the per sentence AER versus the normalized model cost from GIZA++ in the horizontal axis.

Alignment Errors: Word Level Analysis

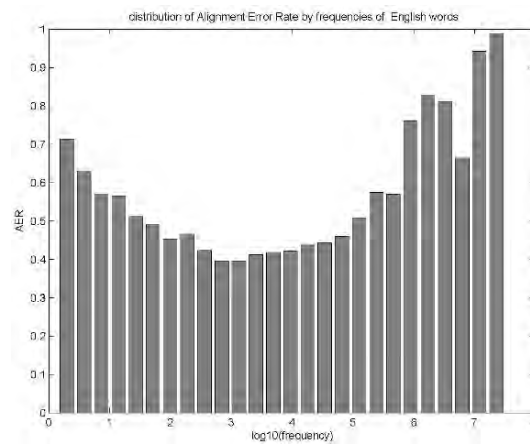


Figure 2.6: Distribution of AER per frequency of English words of the Chinese-English alignment generated by GIZA++ S2T. The vertical axis shows the per word AER versus the log frequency of the English words displayed along the horizontal axis.

The distribution of words in the training corpus typically follows a Zipf curve. This leads to the question if generating correct alignments is more problematic for very low frequency words (as there is not much evidence from the data), or for high frequency words, as they are seen co-occurring with almost every other word. In Figure 2.6 the AER with respect to word frequency is displayed, showing that indeed both high frequency words and low frequency words have higher alignment error rates than the mid frequency words. The word frequencies for this analysis were estimated with the counts over the corpus in Table 2.21.

Chinese Word	Stop Sign “.”			Chinese Word	Determiner “the”		
	TP	FP	FN		Agrmt	FP	FN
,	1176	6540	4114	@NUM	3	19	573
。	10325	2231	2071	美国	0	2	507
“	3	389	9	这	96	200	593
了	0	248	0	的	1	3	367
嗯	1	229	0	政府	0	3	363
的	0	177	1	—	10	54	297
)	0	116	3	国	2	12	293
对	0	110	0	公司	0	0	213
报道	0	98	0	问题	0	4	211
@NUM	1	97	15	世界	0	0	208

Table 2.24: Summarization of alignments of high frequency English words “the” and “.” extracted from Chinese-English alignment generated by GIZA++ S2T

The errors in high frequency words are, of course, more problematic, as they add more to the overall alignment error rate. As we saw in Table 2.22, there are some disparities between the number of errors for the two alignment directions, in particular, the false negatives differ substantially. We, therefore, analyzed the errors for these words in more detail to see, which words contribute most to these errors. Figure 2.7 shows the distribution of the misalignments (fp) and missing links (fn) for the most frequent English words in the GIZA source to target alignments for Chinese. The most frequently misaligned word is the stop sign. On the other side, the word for which the required link is missing most often is the word “the” on the English side. In Table 2.24 we present some examples that illustrate those cases. As we can see from the distribution, the stop sign is misaligned more than half of the time. Other punctuation marks have also high alignment error rates. One conclusion that we can draw from looking at this table is that for Chinese, very frequent words such as the stop sign get often aligned to content words. On the other hand, there are many cases in which the English determiner “the” is missing from the alignment. This kind of behavior could be avoided if we could integrate some

linguistic information into the generative models. Fortunately, there are other alignment models (such as the Discriminative Model), which enable us to do so.

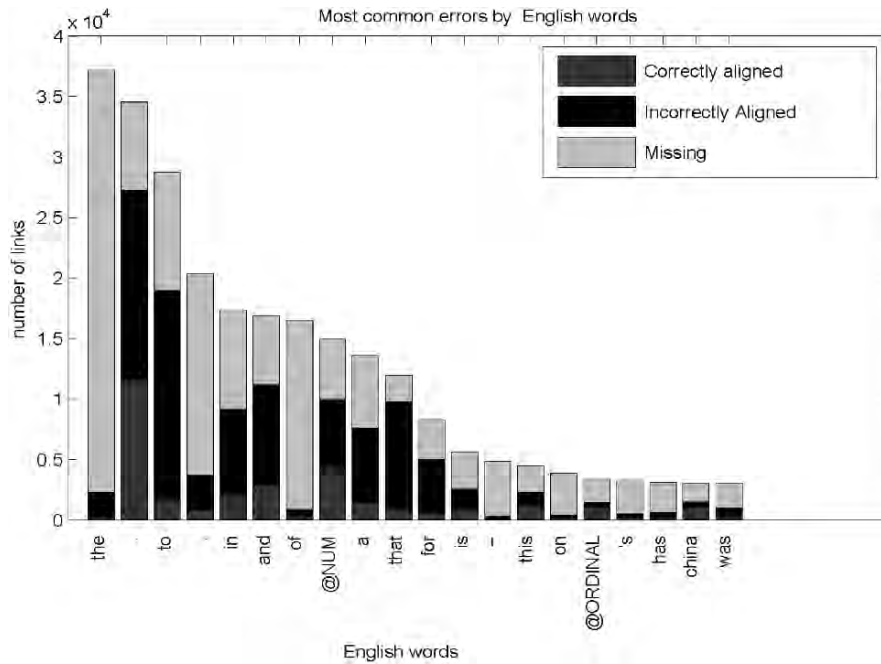


Figure 2.7: Alignment errors for high frequency English words from the Chinese-English alignment generated by GIZA++ S2T

2.3.1.3 Discriminative Word Alignment

In recent years several authors (Moore 2005; Taskar *et al.* 2005; Blunsom and Cohn 2006) proposed discriminative word alignment frameworks and showed that this leads to improved alignment quality. One main advantage of the discriminative framework is the ability to use all available knowledge sources by introducing additional features. In this study, we use the discriminative model presented by Niehues and Vogel (2008), which uses a conditional random field (CRF) to model the alignment matrix.

The structure of the described CRF is quite complex and there are many loops in the graph structure, so the inference cannot be done exactly. Consequently, an approximation algorithm has to be used. For this implementation, the belief propagation algorithm introduced by Pearl (1988) is used. This algorithm is not exact in loopy graphs and it is not even possible to prove that it converges, but by Yedidia *et al.* (2003), it was shown that this algorithm leads to good results.

The weights of the CRFs are trained using gradient descent for a fixed number of iterations. The default method of training CRFs is to maximize the log-likelihood of the correct solution, which is given by a manually created gold standard alignment. However, in our experiments, the tuning process consists on a two step optimization: first we optimize towards maximum likelihood, then towards AER. This sequence has been shown to provide the best results (Niehues and Vogel 2008).

Baseline Features

In the model we are using, there are three different types of factored nodes corresponding to three groups of features. The first group of features is made up of those that depend only on the source and target words and are called local features. For instance, the lexical features, which represent the lexical translation probability of the words, belong to this group. In addition, there are source and target normalized lexical features for every lexicon. The source normalized feature, for instance, ensures that all translation probabilities of one source word to the different target words in the sentences sum up to one. The next group of features is made up of fertility features. They model the probability that a word translates into one, two, three or more words, or does not have any translation at all. The third group contains first order features, which model dependencies between different links. They are used to model relative positions of words, i.e., word reordering (also called distortion) within sentences. For further reference on the detail of these features, please see the work of Niehues and Vogel (2008).

For our experiments the features for the baseline are set as follows: IBM4 lexicons both directions, IBM4 fertilities, IBM4 Viterbi alignments as well as source/target normalization features, identity feature, relative position feature, as well as, the following directions for the first order features: (1,1),(1,2),(2,1),(1,-1),(0,1),(1,0).

New Discriminative Features

One of the purposes of performing the extensive analysis in Section 2.3.1.2 was to recognize the weaknesses of the generative models and, therefore, incorporate new knowledge into the DWA in the form of new features. Such features, along with the existing ones are expected to refine the alignment models and to result in an improvement of alignment quality. To facilitate the integration of such information, we devised a new class-based feature, which scores the GIZA++ alignments that are used for discriminative training, according to the class of the source and target words.

This class-based feature is a local feature and is defined as the conditional probability of a link between words f_j and e_i given the classes of the words $C(f_j)$ and $C(e_i)$; and the alignments provided by GIZA++ in both directions. Using this feature, we can couple any kind of information regarding the class of words (whether they are POS tags, frequency

based classes, etc.) with the alignment information from GIZA++. Then, we use this probability to provide a degree of confidence for the GIZA++ alignments.

This degree of confidence for the GIZA++ alignments has to be estimated according to labeled data, during a feature training phase. This information is then integrated into the DWA as a weighted link feature. The estimation for the class-based feature is obtained through a simple MLE:

$$p(y_{i,j} | \dots) = p(y_{i,j} | C(f_i), C(e_j), L_{s2t}(j, i), L_{t2s}(j, i)) = \frac{|C(f_i), C(e_j), L_{s2t}(j, i), L_{t2s}(j, i), L_{ha}(j, i)|}{|C(f_i), C(e_j), L_{s2t}(j, i), L_{t2s}(j, i)|} \quad (2.9)$$

where $C(f_j)$ is the class of the source word, $C(e_i)$ is the class of the target word, $L_x(j, i)$ a binary function indicating the existence of a link between the words f_j and e_i in a certain alignment set (source to target, target to source, hand aligned).

We tested this kind of feature using a frequency-based classification of the source and target words. That is, we split the words into different classes according to their frequency. These classes are computed in such a way that they have similar number of word-counts. In the current implementation, the number of frequency based classes is set to ten for source and ten for target words.

2.3.1.4 Experiments and Results

In our experiments we wanted to compare the output of the Discriminative Word Aligner (DWA) to that of the Viterbi alignments from GIZA++, as well as, the combined alignment typically used for phrase extraction. As we stated before, the DWA allows one to combine several sources of information. Among them are the Viterbi alignments, the IBM-4 lexicons and IBM4 fertilities and relative position features. For the experiments described in this section, we also tested the frequency-class feature described in the previous section. This feature was trained using the data set previously defined for analysis (Table 2.21). 2552 Arabic-English and 2000 Chinese-English sentences were used for evaluation and the results of the alignment results are shown in Table 2.25.

From these results we can conclude the following: First, discriminative word alignment improves over just heuristically combining the Viterbi alignments from the generative alignment models. This is more significant for Arabic, where the gain is of 4.3 points. Whereas for Chinese is only of 2.6. Adding the frequency feature improves the precision for both Arabic and Chinese, however, leads also to a drop in coverage, giving a modest improvement in AER only for the Chinese to English alignment.

Alignment	Alignment Quality		
	Precision	Recall	AER
Arabic-English			
GIZA S2T	71.9	67.4	30.4
GIZA T2S	64.5	72.7	31.6
combined	68.0	76.5	28.0
DWA - baseline	80.5	72.5	23.7
DWA+Freq-class	83.7	69.6	24.0
Chinese-English			
GIZA S2T	51.7	34.9	58.3
GIZA T2S	66.5	56.9	38.7
combined	68.0	56.3	38.4
DWA - baseline	72.9	57.4	35.8
DWA+Freq-class	75.8	56.7	35.1

Table 2.25: Alignment quality results for the different aligners

2.3.1.5 Conclusions and Future Work

We analyzed word alignments resulting from the standard generative word alignment models. This analysis revealed fundamental problems with those alignment models. Most notably, while all source words are aligned to exactly one target word or the NULL word, a high percentage of the target words are not aligned. Symmetrizing the alignment using heuristics to combine the Viterbi alignments for both directions, while reducing the number of unaligned words, does often not result in any substantial improvement in the alignment quality. On the other side, discriminative word alignment, which can incorporate additional information, can lead to significant improvement over the generative alignment models.

One additional problem with the word alignment remains unexplored in this study: how do changes in word alignment relate to translation quality. Again, as we use heuristics to extract phrase pairs from the word alignment, an improvement in alignment error rate does not necessarily translate into better phrase tables and, thereby, into better translations. To fully benefit from discriminative word alignment the phrase extraction process needs to be adjusted.

2.3.2. Improved Word Alignment Algorithms for Arabic-English and Chinese-English

Authors: Abraham Ittycheriah, Fei Huang, Salim Roukos, and Abhishek Arun

2.3.2.1 Introduction

In word alignment algorithms, we seek a correspondence between words and phrases in a source sequence, $S = [s_1, s_2 \dots s_K]$ and a target sequence of words, $T = [t_1, t_2 \dots t_M]$. In this section, we document recent improvements in the Arabic-English and Chinese-English word alignment algorithms utilizing the basic algorithm presented by

Ittycheriah and Roukos (2005). Word Alignment as a statistical algorithm was first presented by Brown *et al.* (1993). The IBM Models 1-5 have been incorporated into the GIZA++ toolkit, which is the basis for many machine translation approaches (for example see the work of Koehn *et al.* (2007)). Word alignment remains dominated by unsupervised approaches (Brown *et al.* 1993; Vogel *et al.* 1996; Cherry and Lin 2003; Liang *et al.* 2006a); this is largely due to the success of sentence alignment algorithms to produce parallel corpora and, therefore, produce vast quantities of data for unsupervised approaches to word alignment. A survey of word alignment techniques is presented by Och and Ney (2003). Ittycheriah and Roukos (2005) showed that a small quantity of supervised data together with Model 1 statistics from the unsupervised corpora yield significant improvements in alignment accuracy.

Alignments are specified by means of a variable attached to each target element that specifies which one source element it is attached to; this requirement reduces the space of alignments from $2^{|S+1||T|}$ to $|S+1|^{|T|}$. Many-to-many alignments are often warranted in natural language especially in the case of idioms, but initially we ignore these and focus on the remainder of the words and return to this via post-processing mechanisms.⁹

The statistical alignment problem seeks the optimal alignment A^* ,

$$A^* = \operatorname{argmax}_{A \in \mathcal{A}} p(A|S, T) \quad (2.10)$$

where \mathcal{A} is the set of all possible alignment configurations that satisfy the above requirement on alignments. Following the developments of Ittycheriah and Roukos (2005), an alignment can be considered as a series of alignment decisions,

$$p(A|ST) = p(a_i^M | t_1^M, s_1^k) \quad (2.11)$$

$$= \prod_{i=0}^M p(a_i | t_1^M, s_1^k, a_1^{i-1}) \quad (2.12)$$

This is then factored into a transition model and an observation model with a weighting parameter α ,

$$p(A|ST) = \frac{1}{Z} \prod_{i=0}^M p(a_i | a_{i-})^\alpha p(a_i | t_1^M, s_1^k, a_1^{i-})^{1-\alpha} \quad (2.13)$$

where $i-$ refers to all previous decisions up to i .

⁹ We extend the source sequence to have an extra item called the NULL cept (Brown *et al.*, 1993) to which we align spontaneous target words. In our notation from above $|S+1| = K+1$ and $|T| = M$.

Transition Model

The transition model serves as a shape function and is larger for those source positions where it is likely to transition to next in the alignment process. The transition model primarily serves to minimize the distance in terms of source positions between consecutive alignments. Two additional concepts are captured in the transition model:

Frontier

In the alignment process, we align target words from left to right and the frontier is the right-most state (source word) that has a previous target word aligned to it. The frontier notion captures the frequent noun and adjective reversals in Arabic and the long jumps after a “DE” construction in Chinese.

Fertility

The number of target words aligned to a source word determines the fertility of the source word. In this aligner, the translation model is specified as,

$$p(a_i|a_{i-1}) = \frac{1}{Z(a_{i-1})} \left[\frac{1}{\text{dist}(a_i, a_{i-1})} + \frac{1}{\text{ns}(a_i)} \right] \quad (2.14)$$

Where $\text{ns}(i)$ represents the fertility or state visits for state i , $Z(a_{i-1})$ is the normalization constant and

$$\text{dist}(a_i, a_{i-1}) = \min(|a_i - a_{i-1}|, |a_i - f_i|) + \epsilon. \quad (2.15)$$

Here ϵ is a penalty for a zero distance transition and is set to 1 in the experiments below; f_i is the frontier position and a_{i-1} is the previous alignment position. An example alignment for Chinese-English is shown in Figure 2.8. In the example, aligning the period in English has a frontier at source position 11 (because “on the functioning” which occurs earlier in the target string is aligned to position 11) and, therefore, the jump is 1, whereas without the frontier notion, the jump would be 5.

IP-NP	PN	1	g	我们	we	g	1	PRP	NP-S
VP	VV	2	g	认为	believe	g	2	VBP	VP
	PU	3	extra	,	stability	g	3	NN	NP-S-SBAR
	IP-NP-CP-IP-VP-ADV-AD	4	g	稳定	rests	g	4	VBZ	VP
	VP	5	g	取决于	on	g	5	IN	PP
	NP-NP-NR	6	g	巴勒斯坦	the	g	6	DT	NP-NP
	NP	7	g	民主	proper	g	7	JJ	
	DEC	8	g	的	functioning	g	8	NN	
	VP-ADV	9	g	恰	of	g	9	IN	PP
	VP	10	g	当	palestinian	g	10	JJ	NP
	NP	11	g	运作	democracy	g	11	NN	
	PU	12	g	。	.	g	12	.	

Figure 2.8: An Alignment Example

Observation Model

The observation model is formulated using the maximum entropy formulation (Berger *et al.* 1996) and it models the probability that the source and target elements are aligned. The model details are,

$$y = \psi(a_i) \quad (2.16)$$

$$x = [t_1^{i-1}, s_1^k, a_0^{i-}] \quad (2.17)$$

$$p(y|x) = \frac{1}{Z(x)} \exp \sum_i \lambda_i \phi_i(x, y) \quad (2.18)$$

where y is the future, x is the history and $Z(x)$ is the normalizing constant,

$$Z(x) = \sum_{y'} \exp \sum_i \lambda_i \phi_i(x, y') \quad (2.19)$$

and $\phi_i(x, y')$ are binary valued feature functions. The function ψ selects the source word at the position being aligned or an attribute of this position and is the key difference to the traditional formulations of maximum entropy. The future space is the set of source positions in the current source sentence and in order to generalize the features, we parameterize this space by the attributes observed at the source positions. Thus, the partition function, $Z(x)$, is a summation over all attributes observed at source positions that are available for the current decision, a_i . The model parameters, λ_i , are solved using the IIS algorithm (Della Pietra *et al.* 1995) and converges in less than ten iterations. The features investigated so far in this model for each language pair, are presented below.

Arabic Feature Set

- **Word Pairs (Lexical):** Word based features that define the alignment link. These are very similar to the Model 1 type features except that singleton source and target words are classed. The classing of these rare words ensures that the model has support for unseen words.
- **Context:** Lexical items to the left and right of both the source and target positions being aligned.
- **Segmentation:** In Arabic, the words can be segmented into a detailed representation (Lee *et al.* 2003) and these features ask a question about a target word and the morpheme in Arabic. An example is “Al#” and “the”. In addition, in English we use a crude stemmer, which operates on long words (if the word is longer than 6 characters, it keeps the characters from position 3 to 7; for example, on “morpheme” it keeps just “rphe”).
- **Classing (WordNet):** To group English synonyms, this feature type asks the question about the source word and the target WordNet id (Miller 1990).

- **Spelling Feature:** Due to the small quantities of training data, the out-of-vocabulary rate of this aligner is rather high and this feature measures the spelling overlap between the source and target words. Often names and numbers have high overlap in their spellings. For Arabic, we Romanize the word with vowels in order to maximize the overlap with the English word and measure a string kernel distance between the two tokens.
- **Dynamic Features:** These features capture the notion that if we align a preposition or other introductory word, the next words in both languages are likely to be aligned.
- **Distortion Feature:** This feature measures the distance from the previous aligned word; the distance is clipped to the range [-5,5].
- **Target Head Feature:** This feature is a trigram consisting of the current source and target word and its head word. This feature captures only source and target words that occur in the supervised training data, whereas, the M1-proxy feature presented below is more general.
- **M1-proxy Feature.** This feature is designed to align a spontaneous target word with the source word that is aligned to its head target word. For example, in Figure 2.9, consider the article “the” in “the constitutional assembly”, which should be aligned to the same source word (source position 5) as “assembly”. In most sentence pairs, articles occur several times in the same sentence and can be a significant cause of alignment errors. Since the target sentence is being aligned from left to right, the article is aligned before its head word. The M1-proxy feature utilizes Model 1 probabilities in conjunction with a parse tree as an indicator feature function. Note that this feature function provides a look ahead mechanism for the current decision of where to align the article.



Figure 2.9: Arabic Alignment Example (MT03)

In addition to features used in the Arabic aligner, the following features are utilized for Chinese-English word alignment:

- **Character Features:** These features ask questions about the Chinese characters that compose a word and the English word being aligned in the current alignment decision.

- **Spelling Features:** For Chinese, we utilize the Pinyin form of the word to measure the string edit distance to the English word being aligned in the current alignment. **Parse Label Features.** This feature utilizes syntactic information, i.e., the labels of the source and target constituents that cover the source and target words. Because words within a constituent are often translated together, this feature will ensure the words under certain parse tree labels (such as NP) are more likely to be aligned together.

Smoothing the Observation Model

Parallel corpora tend to be significantly larger than the hand aligned data and we utilize the IBM Model 1 probability matrix to smooth the observation model,

$$p(a_i | \dots) = \frac{1}{Z} p_{ME}(a_i | t_1^{i-1}, s_1^k, a_0^{i-})^\beta p_{M1}(s_{a_i} | t_i)^{1-\beta} \quad (2.20)$$

where β controls the amount of smoothing. Higher order models could be used for smoothing, but Model-1 is found to be empirically better than a similar HMM probability matrix. Experiments in varying β are discussed below. During the search below, we require only a score for the observation model and thus drop the normalization Z .

2.3.2.2 Search Algorithm

The search process is a left-to-right beam (Viterbi) search aligning the target sequence to its source counterparts. Since words repeat in both source and target sequences, two hypotheses can only be merged if their alignment histories have been identical over a significant range of target words. This range has to be the maximum length between repeating words and in our experiments we find that a search history size of 30 is optimal. Since the average length of sentences is typically around this number, for most sentences very little merging occurs in the search process. The number of paths is controlled by the beam size, which is set to around 500 paths.

2.3.2.3 Experimental Data

Training data is comprised of the 35K parallel sentences for Arabic and 10.4K sentences for Chinese. Alignment performance is measured using the balanced F-measure (Fraser and Marcu 2007) and, we additionally report precision and recall figures.

Arabic

Although Arabic-English human alignments are done using unsegmented Arabic, recent machine translation systems built for GALE evaluations benefit from doing

segmentation as performed in the Arabic Treebank (ATB). Mapping from unsegmented to segmented Arabic is done by a Model-1 type alignment of the tokens that are segmented. This introduces some error in the supervised corpus, but is small compared to the overall alignment error rate.

Chinese

Chinese word alignments are done at a character level, which allows any segmentation of the source into words and then alignments can be mapped to this word segmentation. This allows the annotation to be independent of the word segmentation decision.

2.3.2.4 Experiments

Testing of the above model is done for Arabic-English on the MT03 test set provided as part of LDC2006G09 with the same projection of unsegmented tokens to segmented tokens as was done for the training material. Chinese-English is tested on 512 sentences drawn randomly from a parallel corpus of over 10M sentence pairs.¹⁰ As baselines, in Table 2.26, we present results of an HMM algorithm (Vogel *et al.* 1996; Ge 2004) as well as GIZA++ alignments (IBM Model 4).

Features

In Table 2.26, the effect of incrementally adding each feature type is shown for Arabic-English. The Word Pairs feature type reduces the F-measure error ($100 - F$) of the HMM aligner by 48% and over the GIZA++ baseline by 37%. Adding the features shows a steady improvement in performance and with all the feature types, the system achieves an F-measure of 89.1 which is an F-measure error reduction of 55% over HMM and 46% over GIZA++. In order to compare to previous published results, on the same 50 sentences of MT03 used by Ittycheriah and Roukos (2005), we obtain an F-measure of 90.2. This result is not directly comparable since it is on segmented input and the number of input tokens is different from the previously published result.

For Chinese-English, Table 2.27 shows the alignment scores when adding the features incrementally. The Word Pairs feature improves the F-measure error over the HMM algorithm by 17% and over GIZA++ by 4.6%. The Character features improve the F-measure by 1.0 point absolute and the parse label feature adds 0.6 point. Utilizing all the features, the F-measure error is reduced by 38.6% over the GIZA++ algorithm.

A further refinement of the word alignment is to remove unlikely alignment links according to a confidence measure of the alignment links (Huang 2009). This leads to higher alignment precision, but lower recall. The link confidence filtering increased the

¹⁰ Both test sets are available from the authors for GALE participants.

precision by 8 points while reducing the recall by 3.5 points and improved the overall F-measure by 1.4 points.

System	#features	P	R	F
HMM		77.0	74.6	75.8
GIZA++		76.8	83.4	80.0
Word Pairs	163K	88.6	86.3	87.4
+Context	175K	89.8	86.3	88.0
+Spelling	175K	90.0	86.6	88.3
+Segmentation	178K	90.1	86.7	88.4
+WordNet	272K	90.1	86.9	88.5
+Distortion	292K	89.2	88.3	88.8
+Dynamic Word	292K	89.2	88.3	88.7
+Target Head	293K	89.3	88.5	88.9
+M1 Proxy	294K	89.4	88.8	89.1

Table 2.26: Arabic-English Word Alignment Performance.

System	#features	P	R	F
HMM		62.7	48.6	54.7
GIZA++		67.0	55.3	60.6
Word Pairs	65K	66.2	59.0	62.4
+Char	92K	67.3	60.1	63.5
+Context	122K	67.3	60.7	63.8
+Spelling	122K	67.2	60.7	63.8
+WordNet	201K	69.0	62.3	65.5
+ParseLabel	219K	69.6	63.0	66.1
+M1 Proxy	219K	70.8	64.4	67.5
Link Confidence Filtering	N/A	79.2	60.9	68.9

Table 2.27: Chinese-English Word Alignment Performance.

Smoothing

Here we study the effect of the relative weight between the transition weight (α) and the observation model and the smoothing parameter (β) between Model 1 and the maximum entropy (ME) model. First, in Figure 2.10 we fix the Model 1 weight to be 0.1 and vary the transition weight from 0 to 0.9. Next, in Figure 2.11 we fix the transition weight to 0.45 and vary the ME weight from 0 to 0.55. The graphs show a stable performance around the optimal thresholds, which set the Transition Model weight to 0.45, Maximum Entropy Model weight to 0.45 and Model 1 weight to 0.1. These values are used in both Arabic-English and Chinese-English.

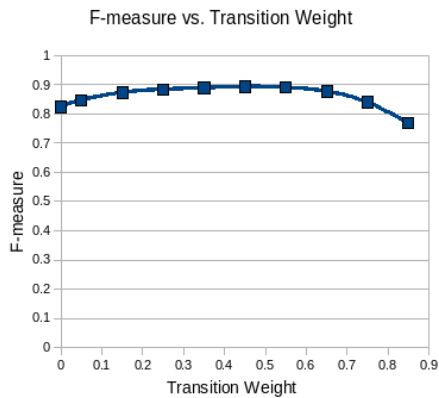


Figure 2.10: Effect of Transition Weight versus the Observation Model Weight for Arabic English.

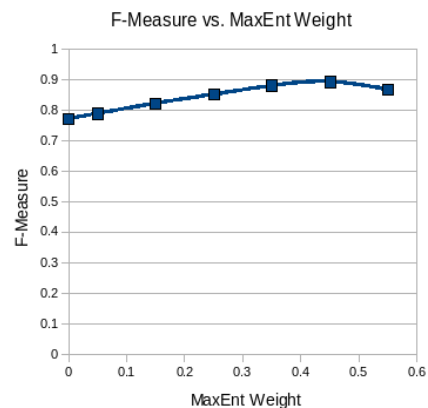


Figure 2.11: Effect of Maximum Entropy Model Weight versus Model 1 Weight.

2.3.2.5 Conclusion

In this section, we have extended the work of Ittycheriah and Roukos (2005) to include several new features and showed its application to the Chinese-English word alignment problem. We showed significant improvement over standard algorithms such as GIZA++'s IBM Model 4. The resulting alignments are utilized in the DTM2 system (Ittycheriah and Roukos 2007) for both Arabic-English and Chinese-English systems.

2.4 Translation Models

2.4.1. Arabic-English Machine Translation Using the Direct Translation Model

Authors: Abraham Ittycheriah and Salim Roukos

2.4.1.1 Direct Translation Models

Statistical machine translation (SMT) takes a source sequence, $\mathbf{S} = [s_1 s_2 \dots s_k]$ and generates a target sequence, $\mathbf{T}^* = [t_1 t_2 \dots t_k]$, by finding the most likely translation given by:

$$\mathbf{T}^* = \operatorname{argmax}_{\mathbf{T}} p(\mathbf{T}|\mathbf{S}) \quad (2.21)$$

This work builds on that of Ittycheriah and Roukos (2007) and in particular we document our system improvements and methodology since GALE Phase 1 in this Section. This section is self-sufficient as we describe in some detail the model and newer experiments. In the seminal paper of Brown *et al.* (1993), Equation 2.21 was factored using the source-channel model into

$$\mathbf{T}^* = \operatorname{argmax}_{\mathbf{T}} p(\mathbf{S}|\mathbf{T}) p^\lambda(\mathbf{T}), \quad (2.22)$$

where the two models, the translation model, $p(\mathbf{S}|\mathbf{T})$ and the language model, $p(\mathbf{T})$, are estimated separately: the former using a parallel corpus and a hidden alignment model and the latter using a typically much larger monolingual corpus. The weighting factor λ is typically tuned on a development test set by optimizing a translation accuracy criterion such as BLEU (Papineni *et al.* 2002). Equation 2.22 is a special case of a Maximum Entropy (log-linear) model described by Papineni *et al.* (1997), called the Direct Translation Model (DTM) with two feature functions. Och and Ney (2002) reintroduced the DTM to machine translation by using a larger number of feature functions (about a dozen or so) along with the use of phrase pairs. The proposed phrase-based DTM is:

$$\mathbf{T}^* = \operatorname{argmax}_{\mathbf{T}} \sum_{m=1}^M \lambda_m h_m(\mathbf{S}, \mathbf{T}), \quad (2.23)$$

where one recognizes the exponential model using m feature functions, $h_m(\cdot)$ and the features weights (scaling factors) are trained using either the maximum entropy principle or more typically by optimizing a translation error metric on a development test set.

Typically, the Direct Translation Models for machine translation have been limited to a dozen or so feature functions. Another key ingredient to the proposed phrase-based DTM model proposed by Och and Ney (2002) is the use of phrases. Briefly, we call a phrase pair of p source words and r target words a “block”. A sentence pair (\mathbf{S}, \mathbf{T}) can be seen as a sequence \mathbf{b}_1^B of B blocks. The phrase-based SMT achieved a significant improvement in translation quality over the source channel model by the use of phrase-pairs in defining the feature functions. The phrase pair library is extracted from a parallel corpus using a word alignment model (such as Model 4 in GIZA or a MaxEnt alignment (Ittycheriah and Roukos 2005)) to align source words to targets words and additional heuristics are used to extract phrase pairs from the aligned parallel corpus. Given a source sentence, we maximize over all possible target sentences and covering of the source sentence by the available phrase library (implicitly defining a block alignment defined by the best scoring hypothesis):

$$\mathbf{T}^* = \operatorname{argmax}_{\mathbf{T}} \sum_{m=1}^M \lambda_m h_m(\mathbf{S}, \mathbf{T}) \quad (2.24)$$

$$= \operatorname{argmax}_{\mathbf{b}_1^B(\mathbf{S})} \sum_{m=1}^M \lambda_m h_m(\mathbf{b}_1^B) \quad (2.25)$$

$$= \operatorname{argmax}_{\mathbf{b}_1^B(\mathbf{S})} \sum_{m=1}^M \lambda_m \sum_{i=1}^B h_m(\mathbf{b}_i | \mathbf{b}_{i-}) \quad (2.26)$$

where $\mathbf{b}_1^B(\mathbf{S})$ covers the input sentence \mathbf{S} and produces the target sentence \mathbf{T} . The feature functions $h(\cdot)$ are decomposed by the sequence of blocks covering the sentence pair (\mathbf{S}, \mathbf{T}) as shown in Equation 2.26, above. For some feature functions such as the block unigram cost and the block Model 1 cost in either direction, $h(\mathbf{b})$ depends only on the current block \mathbf{b} while for other feature functions such as the n -gram language model, they depend on the current block and as many previous blocks as needed to cover the n -gram target word history. Note that a distortion model has been used in some phrase-based systems and that is captured by the distance in words between the last source word of block \mathbf{b}_{i-1} and the first word of the source side of the next block \mathbf{b}_i ; we sometimes refer to this distance as the “jump”, denoted by the variable j . As indicated earlier, current phrase-based systems use a dozen or so feature functions such as the \log unigram frequency of a block \mathbf{b} , the Model 1 scores in both directions for block \mathbf{b} , the n -gram language model score of the target sentence, the number of words in \mathbf{T} , the number of blocks, etc. and the feature weights λ_m are estimated by optimizing an error criterion such as MER (Och 2003). To summarize each of the feature functions $h_m(\cdot)$ is typically a continuous valued function and is typically determined separately by a maximum likelihood estimate (with smoothing in some cases, e.g. the language model) and they are combined with a global set of weights (a small number on the order of a dozen or so). One can see that the phrase-based DTM uses a “block” alignment between source and target sentences and a small number of feature weights.

In our proposed DTM approach (DTM2), we factor the model as:

$$p(\mathbf{T}|\mathbf{S}) = \prod_{i=1}^B p(\mathbf{b}_i | \mathbf{b}_{i-}, \mathbf{S}) \quad (2.27)$$

where \mathbf{b}_{i-} is the sequence of previous blocks producing the partial target sentence left to right and covering a subset of the source sentence \mathcal{S} and we abuse the notation by only allowing blocks \mathbf{b}_i that are consistent with source sentence \mathcal{S} and the partially generated target sentence up to this point. We make explicit the “jump” step between blocks by showing the jump variable in Equation 2.27 as:

$$p(\mathbf{b}_i | \mathbf{b}_{i-}, \mathcal{S}) = p(\mathbf{b}_i, j | \mathbf{b}_{i-}, \mathcal{S}) \quad (2.28)$$

We use a Maximum Entropy model for $p(\mathbf{b}_i | \mathbf{b}_{i-}, \mathcal{S})$

$$p(\mathbf{b}_i | \mathbf{b}_{i-}, \mathcal{S}) = \frac{p_0(\mathbf{b}_i | \mathbf{b}_{i-}, \mathcal{S})}{Z} \exp \sum_m \lambda_m \phi_m(\mathcal{S}, \mathbf{b}_1^{i-1}, j, \mathbf{b}_i) \quad (2.29)$$

where $p_0(\cdot)$ is a prior distribution, Z is a normalizing term and $\phi_m(\mathcal{S}, \mathbf{b}_1^{i-1}, j, \mathbf{b}_i)$ are the features of the model. The feature functions $\phi_m(\cdot)$ are binary valued but are numerous trying to capture detailed aspects of the translation process as described in Section 2.4.1.2 and each has its own weight λ_m ; these feature weights are jointly estimated using the Maximum Entropy Principle (MaxEnt) in contrast to the independently estimated feature functions typically used in today's phrase-based systems. The prior distribution utilized in this work is the normal unigram phrase prior. The definition of the features will be discussed in detail below, however, because we can enumerate all possible futures, this model can be trained using MaxEnt. The set of phrases for each source word or sequence of words then limits the possible target translations. This greatly reduces the number of futures from one of being proportional to the vocabulary size to the number phrases that a source word is part of. Improved Iterative Scaling (Della Pietra *et al.* 1995) is utilized to solve for the feature weights λ_m . MaxEnt allows a very rich definition of features derived from the whole source sentence \mathcal{S} , various analyses of the sentence such as word segmentation, morpheme segmentation, part-of-speech tagging and parsing and the partial target sentence produced at each point, denoted as t^i . We describe our rich set of features in Section 2.4.1.2. The blocks used in our work consist of phrases of lengths (m, n) and were categorized by Ittycheriah and Roukos (2007) into:

1. $n = 0$, source word producing nothing in the target language (deletion block),
2. $m = 0$, spontaneous target word (insertion block),
3. $m = 1$ and $n \geq 1$, a source word producing n target words including the possibility of a variable (denoted by \mathbf{X}) which is to be filled with other blocks from the sentence (the latter case called a discontinuous block)
4. $m \geq 1$ and $n = 1$, a sequence of source words producing a single target words including the possibility of a variable on the source side (as in the French ne...pas translating into not, called multi-word singletons) in the source sequence
5. $m > 1$ and $n > 1$, a non-compositional phrase translation.

Although initial experiments were carried out only on types 1 and 3, the current effort incorporates all types of blocks except Type 2. The phrases are derived from

the alignment and in particular the word position shown in Figure 2.12. Examples of phrases and their categorization is given in Table 2.28.

Word Position	Source Phrase	Target Phrase	Categorization (Type)
1	w#	e_0	Deletion (1)
2	ftHt	opened	Normal (3)
3	mrAkz	the X centers	Normal with Variable (3)
8,9	AlsAEp \$num	\$num	Multi-source word (4)
11,12	Altwqyt AlmHly	local time	Non-compositional (5)

Table 2.28: Complex Phrase Examples

In row 5, “Altwqyt AlmHly” seems to have a simpler word to word alignment, but we are limited to the alignment provided by the word alignment algorithm. The algorithm extracts more complex phrases whenever a simple block extraction fails. Thus, the resulting phrases still form a minimalist set.

S	CONJ	1	extra	w#	the	g	1
	VP	PV	2	g	ftHt	g	2
	NP	SBJ	3	g	mrAkz	g	3
	NP	DET_NOUN	4	g	AlAqtrAE	g	4
	NP	OBJ	5	g	AbwAb	g	5
	NP	POSS@PRON	6	g	+hA	g	6
	NP	ADV	7	g	End	g	7
	NP	NP-NP-DET_NOUN	8	g	AlsAEp	g	8
	NP	NUM	9	g	\$num_(00,06)	g	9
	PP	PREP	10	extra	b#	g	10
	NP	DET_NOUN	11	g	Altwqyt	g	11
	NP	DET_ADJ	12	g	AlmHly	g	12
	PUNC		13	g	.	g	13

Figure 2.12: Arabic Parse Example

2.4.1.2 Features

The features used in this model are all binary questions about the phrase that is being generated and the previous generations. The features can be categorized into the following broad categories:

1. **Lexical Features:** These are internal features to a phrase pair. They examine the source word(s), target word(s) and the jump from the previously translated source word and the size of the source phrase and target phrase. The target size indicates the “fertility” of this phrase and the source size allows the feature to be specific for different phrase sizes. As an example, for the Arabic word “mrAkz” above, the feature when it generates “the” is,

$$\phi(t, j, s) = \begin{cases} 1 & \text{if } s = \text{mrAkz}, t = \text{the}, j = 3 \\ 0 & \text{otherwise} \end{cases} \quad (2.30)$$

where the jump of 3 is obtained because the previous target word is the boundary which is aligned to the source boundary and we need to jump 3 words on the source side to produce the target “the” from the Arabic “mrAkz”.¹¹ The feature when “mrAkz” generates the word “centers” has a jump of -1 since the previous target word comes from the next Arabic word “AlAqtrAE”. The jump captures the complex generation story involved with the variable.

2. **Lexical Context Features:** These exemplify the lexical neighbors of the generation. The previous target words and the neighborhood of the source word are examined.
3. **Arabic Segmentation Features:** An Arabic segmenter is utilized to produce detailed morphemes; these features examine Cartesian product of the Arabic segments together with the English words. Segments which are exclusively identified to only certain English words are restricted to those words by means of a dictionary.
4. **Part-of-Speech Features:** For each language part-of-speech taggers provide additional class information which are used in this feature type.
5. **Coverage Features:** These features examine the state of the source positions surrounding the current generation point. The state can be (a) covered: meaning that this source position has already generated an English phrase, (b) partial: indicating that this source position has generated some text, but is waiting for a variable to be filled, (c) uncovered: indicating that the source position is still open to future generation events.

#	Count	Alpha	Jump	Source	Target	Label	Coverage	Orientation
1	178	1.156	-1	FtHt	opened	NP-SBJ	1	1
2	26	1.323	-2	FtHt	opened	NP-SBJ	1	1
3	142	1.154	1	FtHt	opened	NP-OBJ	0	1
4	18	1.569	1	FtHt	opened	PRT	0	-1

Table 2.29: Features for “ftHt” (Source position 2 in Figure 2.12)

6. **Parse Node Coverage Feature:** A source parser produces parse trees as shown in Figure 2.12. This feature examines if when jumping the current source position there is a parse node that covers the intervening range. The parse node can be in one of four coverage states: (1) uncovered, (2) covered, (3) partial, (4) mixed:

¹¹ Strictly speaking the jump is 2 since the first Arabic word w# uses a deletion block (that produces a null target) and these are handled left to right, but for the sake of simplifying the presentation we do not discuss deletion blocks further.

this state is when the range of source positions of this parse node are in a mixed state of coverage. The feature also indicates the orientation of the parse node, whether it is to the right or left of the current generating source position. In the example shown in Figure 2.12, source position 2 is a verb which is generating the English word “opened”. The Parse Node Coverage feature indicates that the next source position which has an NP-SBJ label above is in a covered state. This feature only fires when the decoder proposes to translate the verb after the NP-SBJ has been covered. The features for this verb are displayed in Table 2.29.

2.4.1.3 Translation Decoder

A beam search decoder similar to phrase-based systems (Tillmann and Ney 2003) is used to translate the source sentence into the target sequence. The decoder is a skip-window width decoder where the source sentence is traversed left to right. Each hypothesized target sequence remembers:

1. Coverage status (both Complete and Partial covered states)
2. The last covered source position

The coverage status indicator is scanned to find the first open source position and a window of source positions is then proposed as generating the next target phrase. As is normally done in phrase based systems, the following meta-feature functions are computed with their weights given in parenthesis:

1. Maximum Entropy model score $p(t_i, j|s)$ (0.35)
2. The phrase pair unigram score (0.01)
3. Model 1 score, $p(s|t)$ (0.2)
4. Language Model score (0.3): This is implemented as a foreground 5-gram LM which encompasses all monolingual, as well as, the target side of the parallel corpus with weight of 0.2; the second component is a domain 4-gram LM, which trained on just the parallel corpus with a weight of 0.1.
5. Word count score (-0.15)

These are weighted in the log domain and added to form the score for this extension. The weights are tuned on the MT06 data set for both Arabic and Chinese. The word count weight is negative indicating a boost in the score for phrases that produce more target words. Word count weight is the only parameter that varies for the languages and in Chinese the system typically produces less target words than is desired and thus, we use a weight of -0.2. Performance is not very sensitive to small changes in the remaining parameters. Since the phrases employed in DTM are simple, a more extensive search is required to find the right ‘re’-ordering of the phrases. For Arabic, the skip parameter is 4 and window width is variable between 5 and 14. The longer window width is applied when the part-of-speech tag of the Arabic word indicates it’s a verb.

GALE Phase 3 Experiments and Analysis

Recent experiments on Arabic-English were carried out on the GALE Dev07 (LDC2007E16) corpus. This corpus has 579 sentences (21K words) in the newswire section and this is utilized as the test data below. As the training data increases with more recent material, older tests are less reliable predictors of performance as a certain amount of contamination occurs from the training data. Hence, here we repeat the experiments carried out by Ittycheriah and Roukos (2007) and also contrast the new PARSE-based features. Also, in GALE the official metric is Human TER by which means that human references are created and system's output are post-edited to match the meaning of this reference and then TER (Snover *et al.* 2006) is computed between the system output and the post-edited system. We utilize a combination of metrics in an effort to get a better correlation to HTER. This combination is essentially equally weighting BLEU and TER which is presented below as $(\text{TER-BLEU})/2$.

Sub-Sampling

For the Arabic-English language pair, we have about 7.8 million sentence pairs. Test sets typically come from more recent material and sub-sampling is a method by which a parallel corpus is derived that is closer to the test set. In order to do this, n -grams up to order 15 are collected from the test set. For each n -gram, we require 20 examples from the larger parallel corpus and for the Dev07 test set we obtain a total of 295,913 sentences on which to train the DTM model. Model 1 is trained on all of the training data. Sub-sampling typically yields 1 to 2 BLEU point improvements over a general system.

Experiments

A feature ablation study is carried out on Dev07 and is presented in Table 2.30. The different feature types have slight variation between test sets, but the general trend is that newer feature sets improve the scores. Of course, features are developed in an error-driven approach and hence, this is not entirely surprising. The baseline phrase decoder is a multi-stack decoder that utilizes n , m phrases and is trained on a similar sub-sampled corpus.

Analysis

- **Component Errors:** Part-of-speech tags and Parse structures are used in the features and when these algorithms make errors, often the final result is impacted.
- **Alignment Errors:** The derivation of the algorithm is predicated on a word alignment between the source and target sentences. Since we have limited the training to just the alignments provided, errors made by the aligner are a significant part of the errors in the translation system. This is in contrast to many approaches which still question the utility of improving alignments.
- **Model Errors:** A fundamental assumption made in this model is that the number of translations for each source sequence can be limited to 15 to 20. This assumption is a poor one for frequent Arabic words which can translate into a number of frequent English phrases.

- Search Errors: Restricting the search to a skip of 4 source words and a window width of up to 14 words is still restrictive and a few sentences require a more significant search.

Feature Types	# offeats	TER	Bleu	(T-B)/2
Baseline Phrase Decoder		41.53	52.90	-5.68
Lexical Features	524528	40.89	52.63	-5.87
+Lexical Context Features	2312573	40.09	54.8	-7.15
+Lexical Trigram Context	3402358	40.12	54.28	-7.08
+Segmentation Features	3574173	40.01	54.33	-7.16
+Variable Features	3584318	40.11	54.20	-7.04
+Coverage Features	371939	40.06	54.30	-7.12
+POS Features	3818413	39.96	55.06	-7.55
+ParseFeats	4063782	39.78	55.37	-7.79

Table 2.30: Dev07 Feature Study.

The errors that the system makes can be categorized into:

- Component Errors: Part-of-speech tags and Parse structures are used in the features and when these algorithms make errors, often the final result is impacted.
- Alignment Errors: The derivation of the algorithm is predicated on a word alignment between the source and target sentences. Since we have limited the training to just the alignments provided, errors made by the aligner are a significant part of the errors in the translation system. This is in contrast to many approaches which still question the utility of improving alignments.
- Model Errors: A fundamental assumption made in this model is that the number of translations for each source sequence can be limited to 15 to 20. This assumption is a poor one for frequent Arabic words which can translate into a number of frequent English phrases.
- Search Errors: Restricting the search to a skip of 4 source words and a window width of up to 14 words is still restrictive and a few sentences require a more significant search.

2.4.1.4 Conclusion and Future Work

We have presented a novel formulation for the translation model. The model incorporates millions of features to learn the neighboring context to guide the search process in choosing the right word choice and order for the translating the source sentence. A feature ablation study shows that the platform is able to incorporate more sophisticated features with a resulting gain in a few automatic metrics. In the future, we intend to incorporate a statistical model for choosing the window of source positions to translate.

2.4.2. Factored Translation Models

Authors: Philipp Koehn and Hieu Hoang

2.4.2.1 Introduction

Phrase-based translation models form a strong baseline for statistical machine translation, especially since they make few assumptions about the training data. However, a number of translation problems, especially handling morphology and reordering are rooted in linguistic properties that are not well addressed by these models. There have been many pursuits motivated by the integration of linguistic knowledge into phrase-based translation models. One promising direction is to follow the structural insight that language is fundamentally recursive and hence translation models should also be structured in the form of recursive (or hierarchical) rule applications (Chiang 2007).

We are concerned here with another direction: the insight that many translation problems require a linguistic vocabulary that goes beyond the notion that language is a sequence (or tree structure) of word tokens. Several properties of words may matter for translation, such as their part-of-speech, their morphological features, or their syntactic or semantic behavior.

To integrate such diverse properties of words into phrase-based translation models, we propose to represent words not as mere tokens from a finite vocabulary, but as vectors of factors. In this section, we outline factored translation models and show how they may be used to address problems such as reordering, rich morphology, syntactic agreement and other characteristics of language that pose problems for translation.

2.4.2.2 Related Work

A wide range of prior work exists on the idea of integrating linguistic information into phrase-based statistical machine translation models.

Morphology — Statistical models for morphology as part of a statistical machine translation system have been developed for inflected languages (Nießen and Ney 2001, 2004). Morphological features may be replaced by pseudo-words (Goldwater and McClosky 2005). Morphological annotation is especially useful for small training corpora (Popovic' *et al.* 2005). For highly agglutinative languages such as Arabic, the main focus is splitting off affixes with various schemes (Habash and Sadat 2006), or by combination of such schemes (Sadat and Habash 2006). A straight forward application of the morpheme splitting idea may not always lead to performance gains (Virpioja *et al.* 2007). Yang and Kirchhoff (2006) present a backoff method that resolves unknown words by increasingly aggressive morphological stemming and compound splitting. Denoual (2007) uses spelling similarity to find the translation of unknown words by analogy. Talbot and Osborne (2006) motivate similar work by reducing redundancy in the input language, again mostly by morphological stemming. Lemmatizing words may improve word alignment performance (Corston-Oliver and Gamon 2004). Using the frequency of stems in the corpus in a finite state approach may guide when to split (El Isbihani *et al* 2006), potentially guided by a small lexicon (Riesa and Yarowsky 2006).

Splitting off affixes may also be a viable strategy when translating into morphologically rich languages (Durgar El-Kahlout and Oflazer 2006). Different morphological analyses may be encoded in a confusion network as input to the translation system to avoid hard choices (Dyer 2007a, b). This approach may be extended to lattice decoding and tree-based models (Dyer *et al.* 2008).

Generating rich morphology — Generating rich morphology poses different challenges. Often relevant information is distributed widely over the input sentence or missing altogether. Minkov *et al.* (2007) use a maximum entropy model to generate rich Russian morphology and show improved performance over using the standard approach of relying on the language model. Such a model may be used for statistical machine translation by adjusting the inflections in a post-processing stage (Toutanova *et al.* 2008). Translation between related morphologically rich related languages may model the lexical translation step as a morphological analysis, transfer and generation process using finite state tools (Tantug *et al.* 2007a). But also, splitting words into stem and morphemes is a valid strategy for translating into a language with rich morphology as demonstrated for English–Turkish (Oflazer and Durgar El-Kahlout 2007) and English–Arabic (Badr *et al.* 2008), and also for translating between two highly inflected languages as in the case of Turkman–Turkish language pairs (Tantug *et al.* 2007b). Translating unknown morphological variants may be learned by analogy to other morphological spelling variations (Langlais and Patry 2007). For very closely related languages such as Catalan and Spanish, translating not chunks of words, but chunks of letters in a phrase-based approach achieves decent results and addresses very well the problem of unknown words (Vilar *et al.* 2007).

Translating tense, case and markers — Schiehlen (1998) analyses the translation of tense across languages and warns against a simplistic view of the problem. Murata *et al.* (2001) propose a machine learning method using support vector machines to predict target language tense. Ye *et al.* (2006) propose to use additional features in a conditional random field classifier to determine verb tenses when translating from Chinese to English. Ueffing and Ney (2003) use a preprocessing method to transform the English verb complex to match its Spanish translation more closely. A similar problem is the prediction of case markers in Japanese (Suzuki and Toutanova 2006), which may be done using a maximum entropy model as part of a treelet translation system (Toutanova and Suzuki 2007), or the prediction of aspect markers in Chinese, which may be framed as classification problem and modeled with conditional random fields (Ye *et al.* 2007). Another example of syntactic markers that are more common in Asian than European languages are numeral classifiers (as in three sheets of paper). Paul *et al.* (2002) present a corpus-based method to generate them for Japanese and Zhang *et al.* (2008a) present a method for generating Chinese measure words. Dorr (1994) presents an overview of linguistic differences between languages, called divergences (Gupta and Chatterjee 2003).

Translation of chunks and clauses — To approach the problem of the translation of a sentence, one strategy is to break up the problem along syntactic lines, be it clauses (Kashioka *et al.* 2003) or syntactic chunks (Koehn and Knight 2002; Schafer and Yarowsky 2003a, b). This strategy also allows for special components for the translation of more basic syntactic elements such as noun phrases (Koehn and Knight 2003) or noun compounds (Tanaka and Baldwin 2003). Owczarzak *et al.* (2006) and Mellebeek *et al.* (2006) break up each sentence into chunks, translating the chunks separately and combine the translations into a transformed skeleton. Hewavitharana *et al.* (2007) augment a phrase-based model with translations for noun phrases that are translated separately.

Syntactic pre-reordering — Given the limitations of the dominating phrase-based statistical machine translation especially with long-distance reordering for syntactic reasons, it may be better to treat reordering in preprocessing by a handcrafted component, which has been explored for German–English (Collins *et al.* 2005), Japanese–English (Komachi *et al.* 2006), Chinese–English (Wang *et al.* 2007) and English–Hindi (Ramanathan *et al.* 2008). Zwarts and Dras (2007) point out that translation improvements are due to both a reduction of reordering needed during decoding and the increased learning of phrases of syntactic dependents. Nguyen and Shimazu (2006) also use manual rules for syntactic transformation in a preprocessing step. Such a reordering component may also be learned automatically from parsed training data, as shown for French–English (Xia and McCord 2004), Arabic–English (Habash, 2007b) and Chinese–English (Crego and Mariño 2007). The latter work encodes different orderings in an input lattice to the decoder. Li *et al.* (2007) propose a maximum entropy pre-reordering model based on syntactic parse trees in the source language. It may be beneficial to train different such pre-reordering models for different sentence types (questions, etc.) (Zhang *et al.* 2008b). Preprocessing the input to a machine translation system may also include splitting it up into smaller sentences (Lee *et al.* 2008).

POS and chunk-based pre-reordering — Reordering patterns may also be learned over part-of-speech tags, allowing the input to be converted into a reordering graph (Crego and Mariño, 2006) or enabling a rescoring approach with the patterns as features (Chen *et al.* 2006). The reordering rules may also be integrated into an otherwise monotone decoder (Tillmann 2008). Such rules may also be used in a separate reordering model. Such rules may be based on automatic word classes (Costa-Jussa` and Fonollosa, 2006; Crego *et al.* 2006), which was shown to outperform part-of-speech tags (Costa-Jussa` and Fonollosa 2007), or they may be based on syntactic chunks (Zhang *et al.* 2007a, b; Crego and Habash 2008). Scoring for rule applications may be encoded in the reordering graph, or done once the target word order is established, which allows for rewarding reorderings that happened due to phrase-internal reordering (Elming 2008a, b).

Syntactic reranking — Linguistic features may be added to an n -best of candidate translation to be exploited by a reranking approach. This was explored by Och *et al.* (2004) and by Koehn and Knight (2003) for noun phrases. Sequence models trained on part-of-speech tags (Bonneau Maynard *et al.* 2007), or CCG supertags (Hassan *et al.*

2007), may be also used in reranking. To check if the dependency structure is preserved during translation, the number of preserved dependency links or paths may be a useful feature (Nikoulina and Dymetman 2008).

2.4.2.3 Definition of Factored Models

Factored translation models integrate additional linguistic markup at the word level. Each type of additional word-level information is called a **factor**. The type of information that may be useful could consist of word, lemma, part-of-speech, morphology, word class, etc. The translation of lemma and morphological factors separately would help with sparse data problems in morphologically rich languages. Additional information such as part-of-speech may be helpful in making reordering or grammatical coherence decisions. The presence of morphological features on the target side allows for checking agreement within noun phrases or between subject and verb.

As we argue above, one example to illustrate the short-comings of the traditional surface word approach in statistical machine translation is the poor handling of morphology. Each word form is treated as a token in itself. This means that the translation model treats, say, the word house completely independent of the word houses. Any instance of house in the training data does not add any knowledge to the translation of houses.

In the extreme case, while the translation of house may be known to the model, the word houses may be unknown and the system will not be able to translate it. While this problem does not show up as strongly in English — due to the very limited morphological inflection in English — it does constitute a significant problem for morphologically rich languages such as Arabic, German, Turkish, Czech, etc.

Thus, it may be preferable to model translation between morphologically rich languages on the level of lemmas and, thus, pooling the evidence for different word forms that derive from a common lemma. In such a model, we would want to translate lemma and morphological information separately and combine this information on the output side to ultimately generate the output surface words.

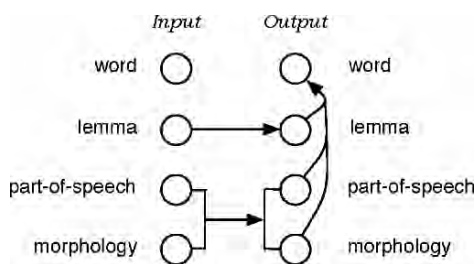


Figure 2.13: Example factored model: morphological analysis and generation, decomposed into three mapping steps (translation of lemmas, translation of part-of-speech and morphological information, generation of surface forms).

Such a model can be defined in a straight forward way as a factored translation model. See Figure 2.13 for an illustration. Note that factored translation models may also

integrate less obviously linguistic annotation such as statistically defined word classes, or any other annotation. For further details, refer to the work of Koehn and Hoang (2007).

Training Factored Models

The training of factored models follows closely the training method for traditional phrase-based models. The training data (a parallel corpus) has to be annotated with the additional factors. For instance, if we want to add part-of-speech information on the input and output side, we need part-of-speech tagged training data. Typically, this involves running automatic tools on the corpus, since manually annotated corpora are rare and expensive to produce. Next, we establish a word alignment for all the sentences in the parallel training corpus using standard methods.

Each mapping step forms a component of the overall model. From a training point of view this means that we need to learn translation and generation tables from the word-aligned parallel corpus and define scoring methods that help us to choose between ambiguous mappings.

Phrase-based translation models are acquired from a word-aligned parallel corpus by extracting all phrase-pairs that are consistent with the word alignment. Given the set of extracted phrase pairs with counts, various **feature functions** are estimated, such as conditional phrase translation probabilities based on relative frequency estimation or lexical translation probabilities based on the words in the phrases.

The models for the translation steps may be acquired in the same manner from a word-aligned parallel corpus. For the specified factors in the input and output, phrase mappings are extracted. The set of phrase mappings (now over factored representations) is scored based on relative counts and word-based translation probabilities. See Figure 2.14 for an illustration.

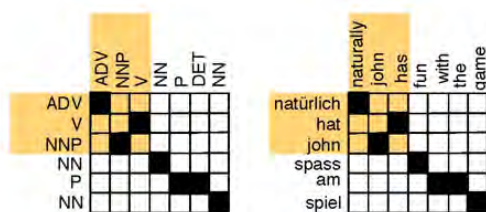


Figure 2.14: Training: Extraction of translation models for any factors follows the phrase extraction method for phrase-based models.

The generation models are probability distributions that are estimated on the output side only. In fact, additional monolingual data may be used. The generation model is learnt on a word-for-word basis. For instance, for a generation step that maps surface forms to part-of-speech, a table with entries such as (*fish*, *NN*) is constructed. As feature functions we may use the conditional probability distributions, e.g., $p(NN/fish)$, obtained by maximum likelihood estimation.

An important component of statistical machine translation is the language model, typically an n -gram model over surface forms of words. In the framework of factored translation models, such sequence models may be defined over any factor, or any set of

factors. For factors such as part-of-speech tags, building and using higher order n -gram models (7-gram, 9-gram) is straight forward.

2.4.2.4 Applications

The framework of factored translation models has many applications, which we outline in this section. Additional factors may be supplied to the input and the output. They may be used in the reordering model or the translation model.

Enriching the Input

When tackling language pairs such as Chinese-English, we often have to create output words and phrases from input that is under specified. Chinese is less richly annotated than English. For instance, it lacks determiners, plural/singular distinction in nouns and verb tenses.

This annotation is not strictly required to bring across the same meaning, since the information may be provided by other means, for instance by mentioning the number of objects or time of the event a few sentences earlier in the document. During the translation process with today's models, however, we do need the information in a very local context when translating a word or a phrase.

We may obtain the needed information from the context and add it as additional factor at the word level. For instance, by co-reference resolution, we can connect up the mentions of an entity and distribute information about singular/plural throughout the entity chain.

We explored the use of factored translation models in such a scenario (Avramidis and Koehn 2008) to add annotation for verbs and noun phrases and showed improvements in selecting the correct morphological variants in the output (reducing verb translation errors from 19% to 5.4% and noun case translation errors from 9% to 6%).

Enriching the Output

A key component of statistical machine translation systems is the language model, which is typically an n -gram model. Since language models of order 5 are considered large, this limits the reach of the component that ensures fluent output to a relatively small local window. Especially for morphologically rich languages, even a 5-gram language model does not have sufficient coverage and frequently backs off to lower order models.

Factored language models (Bilmes and Kirchhoff 2003) were proposed to address these challenges by allowing backoff to more robust histories, such as part-of-speech tags. In the factored translation model framework, we may use sequence models over specific factors, such as part-of-speech tags that are not strictly language models, but serve the same purpose: ensuring fluent output of the translation system.

Sequence models over part-of-speech tags may use much larger windows (we used 7-gram and 9-gram models), due to the much smaller vocabulary, the Penn tree bank tag set has 46 different tags. Such longer-range models ensure grammatical coherence beyond what surface word models are able to address.

The use of additional factors has been explored in various ways. The approach has been shown to be successful for integrating part-of-speech tags (Koehn and Hoang 2007),

word class factors (Shen *et al.* 2006), CCG super-tags (Birch *et al.* 2007), morphological tags (Koehn and Hoang 2007; Badr *et al.* 2008), and shallow syntactic tags (Cettolo and Bertoldi 2008). More complex use of additional output factors have been explored as well (Bojar 2007).

The additional output factors may be either generated by a factored model consisting of a translation step (creating the output word) and a generation step (creating the additional output factor), or by a factored model that creates all output factors in one step. Our experiments have shown that the latter configuration typically performs better.

Factored Reordering

The problem of reordering is recognized as one of the hardest problems when translating between syntactically different languages. Much of the motivation for grammar-based translation models comes from the fact that the output from phrase-based systems is often ungrammatical due to failure to place words in the right output language order.

Reordering between languages is often best described using syntactic concepts: In Arabic-English translation, the verb has to be moved after the subject. In Chinese-English translation, prepositional phrases have to be moved after the noun phrase they modify. Hence, the idea is to use syntactic properties such as part-of-speech tags or syntactic chunk labels during translation.

The lexicalized reordering model (Tillmann 2004; Koehn *et al.* 2005) commonly used in phrase-based models provides an avenue for using linguistic information during reordering. However, instead of using the lexical identity of phrases to inform reordering decisions, we may use their part-of-speech tags.

Part-of-speech tags may inform larger reordering patterns, or reordering templates. For instance, we may find a useful 5-word pattern to reorder words based on their part-of-speech tags, but not the underlying 5-word phrase translation. This motivates the **factored template model** (Hoang and Koehn 2009). This variant of factored translation models allows the use of longer phrases for some factors. In essence, the part-of-speech template drives midrange reordering decisions of the traditional phrase-based translation model.

Morphological Analysis and Generation

Our motivating example was to deal with morphological rich languages the same way rule-based systems deal with them: We want to translate word lemmas and their morphological information separately, thus having a more general model that allows us to increase coverage and robustness.

Our experiments in this area (Koehn and Hoang 2007) have shown two important points. First, enabling the translation of morphological variants not observed in training, but explainable in terms of its morphological analysis, increases coverage and reduce the number of unknown words. Second, relying only on a morphological analysis and generation model leads to a worse performance. The reason for this lies in part in increased decoding complexity and thus, more room for search errors, but also in the many additional independence assumptions that are introduced. In fact, the model, as shown in Figure 2.13 does not use the source word at all anymore, just its morphological

analysis. Our experiments have shown that we ultimately achieve gains in performance by using alternative decoding paths, which means allowing the construction of translation options both through the surface translation model and the morphological model.

Basic models that treat words as tokens work best, if these tokens have been observed frequently in training. For rare or unseen words, the problem of data sparsity does often not allow for correct estimation of translation rules and thus, the morphological model is able to help out.

This leads us to define backoff models for translation. Just as language models back off from higher n -gram models to lower n -gram models, we may back off from one translation model to another, if the first one has no coverage. We already reported on methods that use alternative decoding paths, i.e., may chose between different translation models. Backoff implies a sequence of translation models, where the lower-priority models are only used in the higher-priority models do not apply.

The experience in language models has shown the advantage of interpolated backoff. It may always be better to include estimates from lower-order language models. We may handle backoff translation models the same way, but as in language modeling, the interpolation calculations may be done during training, not decoding. We will report in future papers on experimental results with backoff models.

2.4.2.5 Conclusion

We introduced factored translation models, a framework to use additional annotation at the word level to improve statistical machine translation models in two ways: (1) by allowing more general models that are estimated on richer statistics, such as lemma translation models, part-of-speech or morphological sequence models, and (2) by introducing additional linguistic knowledge, which may be exploited, for instance, during reordering.

We have shown that the framework allows for improvements through enriching the input, enriching the output, richer reordering models and translation models based on morphological analysis.

We plan in future work to apply the idea of factored translation models to tree-based approaches, thus creating a formalism that could be described as probabilistic synchronous unification grammar.

2.4.3. Soft Syntax Features and Other Extensions for Hierarchical SMT

Authors: Daniel Stein, David Vilar and Hermann Ney

2.4.3.1 Introduction

Translation based on word groups (so called phrases) has proven to be an effective approach to statistical machine translation and a great improvement over the initial word-based approaches first presented in the seminal IBM paper of Brown *et al.* (1993). However, these models are still limited. They do not consider long range dependencies, as the phrases must be contiguous both in the source and the target language and the

reordering of the phrases is not directly modeled and is normally applied in a heuristic way.

The hierarchical phrase based translation model (Chiang 2007) addresses these issues by allowing “gaps” in the phrases. In this way the reorderings are integrated in the translation process and non-contiguous word groups can be translated in a consistent way.

In this work, we briefly review the hierarchical phrase-based approach and explore various refinements to the phrase extraction algorithm and the effect on the final translation quality. Additionally, we add syntax information to the extracted phrases, and computing features, which measure how well the extracted phrases correspond to linguistic structures. We expect that including this additional information will help the translation process selecting the most appropriate translation units. This becomes of greater importance for the hierarchical model, as the extraction process allows for a high number of phrases, most of them originating due to the high flexibility of the extraction algorithm and of limited use for the translation.

The advantages of our approach in comparison to other recent syntax-based approaches is, on the one hand, its simplicity, and thus, flexibility, which allows it also to be integrated with standard phrase-based approaches, and on the other hand, the possibility to use syntax information both on the source and target languages.

2.4.3.2 Related Work

The hierarchical phrase based approach was first presented by David Chiang in 2005 (Chiang 2005), and further detailed in 2007 (Chiang 2007). Already in 2005, Chiang proposes the use of syntactic information together with his new hierarchical approach, but without success.

Some recent publications have shown that the use of syntax for translation achieves significant improvements. One prominent example is the work by the ISI group (e.g., (Galley *et al.* 2004; Marcu *et al.* 2006)). This work departs from the standard phrase-based approach by defining new translation units and extraction procedures, but the investigators try to still keep the advantages of phrase-based translation (DeNeefe *et al.* 2007).

Our work differs from the above mentioned mainly in that we extract the syntactic information already at the training phase and it is integrated in the search process as an additional model in the base log-linear combination that underlies most state-of-the-art statistical machine translation systems. Therefore, no modification of the search algorithms is needed and we can also make use of syntactic information for both languages, source and target. Most of the previous work was limited only to the target language side, as the correspondences between the syntactic structures of both languages are hard to define.

2.4.3.3 Hierarchical Phrase Based Translation

Details of the system are described by Vilar *et al.* (2008). The extraction process starts with the same phrase-extraction procedure as in the standard phrase-based translation, i.e., sequences of source and target words that are aligned only to each other.

We then proceed to generate new rules as follows: for each of the extracted phrase pairs, smaller sub-phrases are sought. If found, the corresponding parts are substituted by a non-terminal and linked together by the relation. This process is then iterated with the new extracted rules, until the desired maximum number of non-terminals is achieved. Note however, that only standard phrases are made into gaps.

For phrase-based training, the phrases get scored by relative frequencies, whereas for the hierarchical phrases, the counts of the originating standard phrases get distributed among all the generated hierarchical rules. These scores are computed for the translation directions source-to-target and target-to-source, which get combined log-linearly with additional IBM1-like word level scores at the phrase level, word and phrase penalty scores at generation time.

The decoding process is basically a parsing of the source sentence according to the defined grammar, keeping track of the target language translation contexts in order to compute language model scores during the translation process. The parsing is carried out via the CYK+ algorithm (Chappelier and Rajman 1998), which allows for a CYK-style parsing without transformation of the original grammar. The forest of possible parses is compactly represented in a hypergraph structure. This hypergraph gets traversed in a top-down manner, generating the target language in the process, including language model computations. This pass is done in a lazy way, using the cube growing algorithm. The details can be found in the work of Huang and Chiang (2007).

2.4.3.4 Experiments

In this section we present experimental results for the methods introduced in Vilar *et al.* (2008) on the GALE 2008 development and test data sets. A parsed subset of the training data was kindly provided by the University of Maryland (Huang *et al.* 2008) and used for the training of the syntax components of our system. The statistics of the parsed data can be found in Table 2.31.

	Chinese	English
number of running sentences	671701	
number of running words	2905698	21603468
parsed categories	59	-

Table 2.31: Statistics of the parsed data

We analyze the effect of different extraction heuristics (a binary feature activated for hierarchical phrases, an extended glue rule, ...), as well as, different approaches to include syntax information in the form of soft-syntax features. For more details refer to the above-mentioned paper.

The results on the whole corpora are presented in Table 2.32. It can be seen from this table that the performance of the hierarchical system can be improved by incorporating additional information. The inclusion of the extended glue rule improved the system performance, however, the other heuristic features do not seem to help in the translation process, despite being beneficial for other corpora. The syntactic features show mixed results. While the binary feature seems to help improve the overall quality of the

translation, the other smoothing methods seem to yield over fitting on the development corpus.

	dev		Test	
	BLEU	TER	BLEU	TER
base	24.6	64.5	24.1	64.2
glue2	25.2	64.2	24.5	64.1
all	25.1	64.2	24.2	64.2
hierarch	24.4	64.4	23.8	64.1
paste+hierarch	24.4	64.7	23.7	64.2
paste	24.7	64.3	24.1	64.5
binary	25.1	64.3	24.4	64.1
linear	24.6	65.1	23.8	64.0
log	23.8	65.7	23.2	65.9
relative	24.8	64.4	24.0	64.7

Table 2.32: Results on the GALE dev- and test data. Training was conducted with help of only partially parsed corpora.

The combination of heuristic and syntactic features helps in comparison to the baseline, but, regarding the improvements by the other scores, was expected to be higher. We believe this to be the result of shortcomings of the Downhill Simplex Method used for optimization, since a total 15 different scores have to be optimized. This is probably bigger than the maximum number of variables that can be estimated reliably using this method, a finding consistent with that of Chiang *et al.* (2008). We expect that this effect can be avoided by switching to other optimization techniques like the Margin Infused Relaxed Algorithm (MIRA) and will conduct these experiments in future work.

	BC		BN		NW		WT	
	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER
base	25.1	61.3	25.9	62.0	25.7	62.1	19.6	65.9
glue2	24.8	61.4	26.2	62.0	26.4	61.7	19.8	65.7
all	24.1	61.4	26.0	61.8	26.2	62.0	19.4	66.0
hierarch	24.1	61.5	25.2	62.5	25.6	61.9	19.4	66.0
paste+hierarch	23.8	61.7	25.5	62.1	25.8	61.8	19.2	66.0
paste	24.3	61.6	25.6	62.1	26.2	62.0	19.5	65.9
binary	24.6	61.5	25.9	62.1	26.4	61.5	20.1	66.2
linear	24.0	62.0	25.3	62.9	25.8	62.7	19.3	65.7
log	23.4	62.7	24.7	63.6	25.1	63.3	18.8	66.7
relative	24.5	61.7	25.7	62.3	25.7	62.0	19.6	67.6

Table 2.33: Results on the GALE test data, split by genre

Table 2.33 shows the results for each of the genres composing the test corpus. The biggest improvement using the binary syntactic feature can be seen on the newswire data, followed by the webtext data. This is to be expected, as the broadcast news and especially, the broadcast conversation data are more speech-like data and the sentences tend to have a poorer grammatical structure.

reference	Well I think it's hard to say . Why ?
baseline	This i feel that we are very hard to say . Why ?
syntactical	I think that now it is difficult to say . Why is it so ?

reference	Therefore , we would like to further intensify law enforcement .
baseline	We would like to further increased enforcement .
syntactical	We would like to further increased the intensity of law enforcement .

Reference	The reason is that people are afraid to spend their money . Is that right ?
baseline	This reason is that people rich do not want to spend , right ?
syntax	This reason that is to say , people dare not to spend money , right ?

Figure 2.15: Sample sentences with changes due to syntactical feature

Figure 2.15 shows some example sentences where the translation using the syntactic features was improved.

2.4.3.5 Conclusions

We have analyzed the effect on translation quality of different extraction heuristics for a hierarchical phrase-based translation system. We also have shown how to include syntactic information at the training phase. In this way, we can include syntactic information from both languages at translation time without the need of modification on a decoder. Experiments were reported on the GALE 2008 task, where some syntactic features helped to improve the translation quality. The flexibility of this method also allows it to be included in other systems such as standard phrase-based systems or even syntax-oriented systems, which normally take the target syntax into account. Including these features the source syntax can also be involved in the translation process.

2.4.4. Two String-to-Tree Translation Models

Authors: Steve DeNeeffe, Libin Shen, Kevin Knight, Daniel Marcu, Wei Wang, Ralph Weischedel & Jinxi Xu

2.4.4.1 Introduction

String models are popular in statistical machine translation (MT). Approaches include word substitution systems (Brown *et al.* 1993), phrase substitution systems (Koehn *et al.* 2003; Och and Ney 2004), and synchronous context-free grammar systems (Wu and Wong 1998; Chiang 2007), all of which train on string pairs and seek to establish connections between source and target strings. By contrast, explicit syntax approaches seek to directly model the relations learned from parsed data, including models between source trees and target trees (Gildea 2003; Eisner 2003; Melamed 2004; Cowan *et al.* 2006), source trees and target strings (Quirk *et al.* 2005; Huang *et al.* 2006), or source strings and target trees (Yamada and Knight 2001; Galley *et al.* 2004).

It is unclear which of these pursuits will best explain human translation data, as each has advantages and disadvantages. A strength of phrase models is that they can acquire all phrase pairs consistent with computed word alignments, snap those phrases together easily by concatenation and reorder them under several cost models. An advantage of

syntax-based models is that outputs tend to be syntactically well formed, with reordering influenced by syntactic context and function words introduced to serve specific syntactic purposes.

This section has two parts. In the first part (Section 2.4.4.2), we present a comparison of phrase-based and syntax-based rule extraction methods and phrase pair coverage. Additionally, we do a careful study of several syntax-based extraction techniques, testing whether (and how much) they affect phrase pair coverage and whether (and how much) they affect end-to-end MT accuracy. For this comparison, we choose a previously established statistical phrase-based model (Och and Ney 2004) and a previously established statistical string-to-tree model (Galley et. al. 2004).

In the second part (Section 2.4.4.3), we discuss the string-to-dependency-tree translation model, a syntactic extension of the hierarchical phrase-based model (Chiang 2005) that produces target dependency structures. This model represents the source side of a translation rule as a string and the target side as a dependency structure. It incorporates a dependency LM as a feature in decoding in order to exploit long-distance word relations, which cannot be captured by a traditional n -gram language model.

2.4.4.2 Improving Syntax-Based MT with Phrase-Based MT

Syntax-Based MT

In this section, we describe our syntax-based translation system. This system accepts a foreign string (e.g., Chinese or Arabic) as input and it searches through a multiplicity of English tree outputs, seeking the one with the highest score (see Figure 2.16). The string-to-tree framework is motivated by a desire to improve target language grammaticality. For example, it is common for string-based MT systems to output sentences with no verb. By contrast, the string-to-tree framework forces the output to respect syntactic requirements, for example, if the output is a syntactic tree whose root is S (sentence), then the S will generally have a child of type VP (verb phrase), which will in turn contain a verb.

Another motivation is better treatment of function words. Often, these words are not literally translated (either by themselves or as part of a phrase), but rather they control what happens in the translation, as with case-marking particles or passive-voice particles. Finally, much of the reordering we find in translation is syntactically motivated and this can be captured explicitly with syntax-based translation rules.

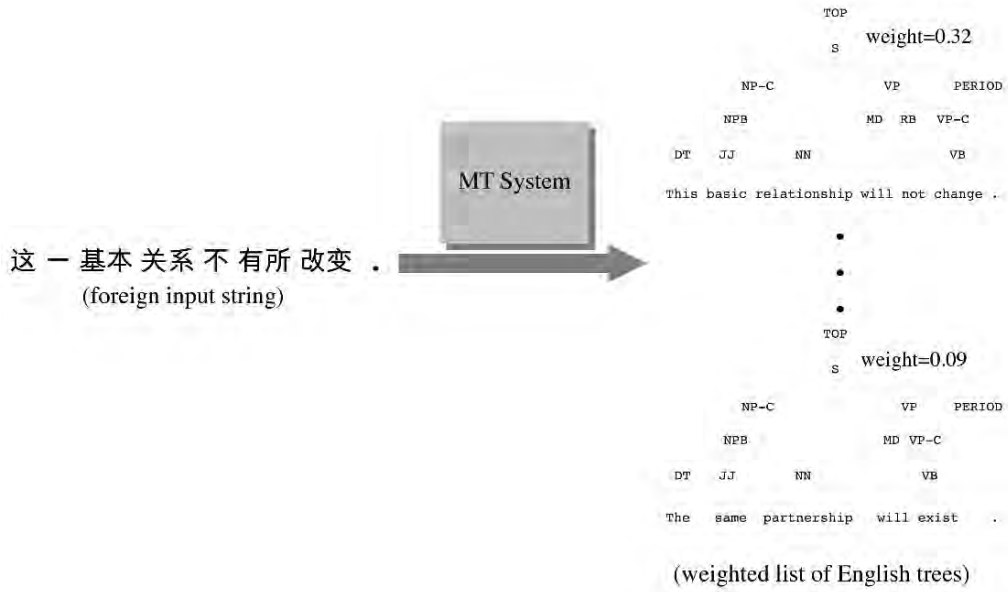
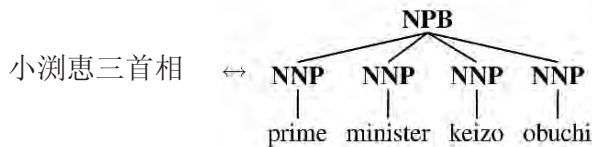


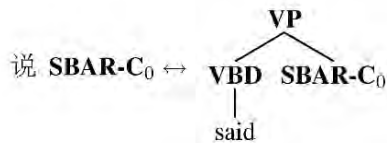
Figure 2.16: the syntax-based translation system accepts a foreign string and produces a set of candidate translation trees with confidence weights

Here are some examples of translation rules and the roles that they play in the translation of a sentence:

- phrase pairs with syntax decoration:

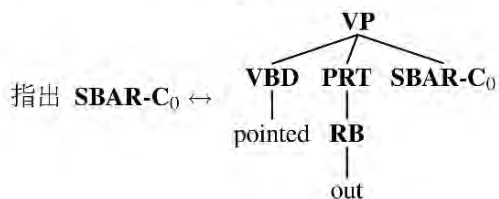
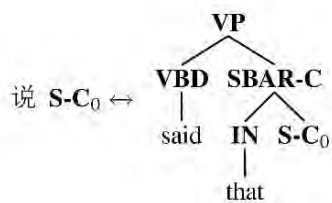


- extra contextual constraints:

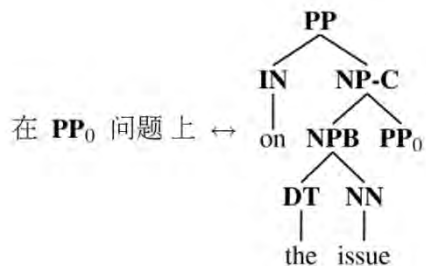
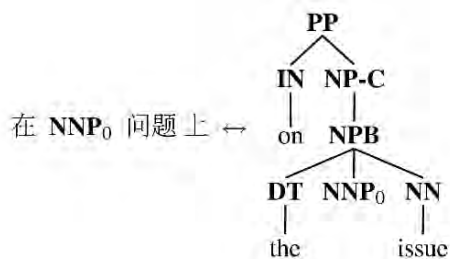


(this rule translates 说 to said, but only if some Chinese sequence to the right of 说 is translated into an SBAR-C)

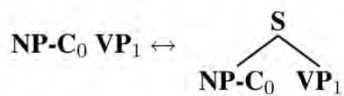
- non-constituent phrases:



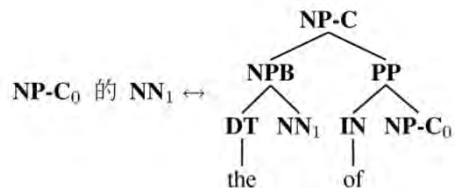
- non-contiguous phrases, effectively “phrases with holes”:



- purely structural (no words):



- re-ordering:



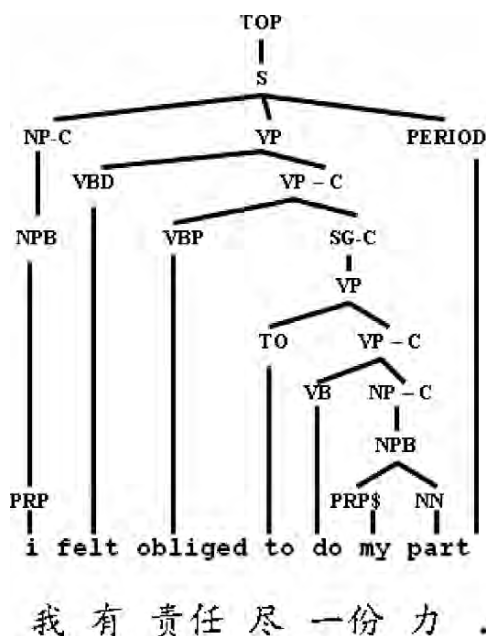


Figure 2.17: a syntax-based training example

To prepare training data for such a system, we begin with a bilingual text that has been automatically processed into segment pairs. Our sentence splitting method ensures that the segments be single sentences on the English side, while the corresponding Chinese segments may be sentences, sentence fragments, or multiple sentences. We then parse the English side of the bilingual text using a re-implementation of the Collins (2003) parsing model, which we train on the Penn English Treebank (Marcus *et al.* 1993). Finally, we word-align the segment pairs according to IBM Model 4 (Brown *et al.* 1993). Figure 2.17 shows a sample \langle tree, string, alignment \rangle triple.

We build two generative statistical models from this data. First, we construct a smoothed n -gram language model (Kneser and Ney 1995; Stolcke 2002) out of the English side of the bilingual data. This model assigns a probability $P(e)$ to any candidate translation, rewarding translations whose sub-sequences have been observed frequently in the training data.

Second, we build a syntax-based translation model that we can use to produce candidate English trees from Chinese strings. Following previous work in noisy-channel SMT (Brown *et al.* 1993), our model operates in the English-to-Chinese direction. We envision a generative top-down process by which an English tree is gradually transformed (by probabilistic rules) into an observed Chinese string. We represent a collection of such rules as a tree-to-string transducer (Knight and Graehl 2005). In order to construct this transducer from parsed and word-aligned data, we use the GHKM rule extraction algorithm of Galley *et al.* (2004). This algorithm computes the unique set of minimal rules needed to explain any sentence pair in the data. This is, in effect, a non-overlapping tiling of translation rules over the tree-string pair. If there are no unaligned words in the source sentence, this is a unique set. Figure 2.18 shows the set of seven

minimal rules extracted from the example (tree, string, word alignment) triple in Figure 2.17 if the unaligned 力 is ignored. This set, ordered into a tree of rule applications, is called the derivation tree of the training example. Unlike the ATS model described next, there are no inherent size limits, just the constraint that the rules be as small as possible for the example.

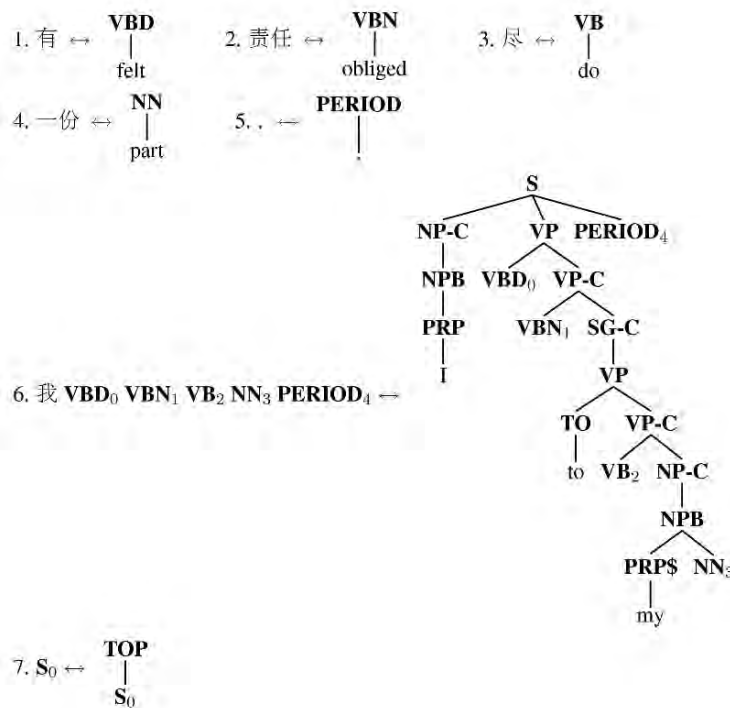
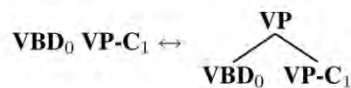
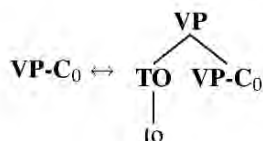


Figure 2.18: rules extracted from the training example in Figure 2.17 (ignoring unaligned Chinese words)

Notice that rule 6 is rather large and applies to a very limited syntactic context. The only constituent node that covers both *i* and *my* is the *S*, so the rule rooted at *S* is extracted, with variables for every branch below this top constituent that can be explained by other rules. Note also that *to* becomes a part of this rule naturally. If the alignments were not as constraining (e.g., if *my* was unaligned), then instead of this one big rule many smaller rules would be extracted, such as structural rules, e.g.,

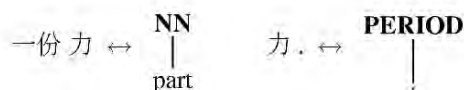


and function word insertion rules, e.g.,



We ignored unaligned source words in the example above. Galley *et al.* (2004) attach the unaligned source word to the highest possible location, in our example, the S. Thus it is extracted along with our large rule 6, changing the source language sequence “我 VBD_0 VBN_1 VB_2 NN_3 力 PERIOD_4 ”. This treatment still results in a unique derivation tree no matter how many unaligned words are present.

In the work of Galley *et al.* (2006), instead of a unique derivation tree, the extractor computes several derivation trees, each with the unaligned word added to a different rule such that the data is still explained. For example, for the tree-string pair in Figure 2.17, 力 could be added not only to rule 6, but alternatively to rule 4 or 5, to make the new rules.



This results in three different derivations, one with the 力 character in rule 4 (with rules 5 and 6, as originally shown), another with the 力 character in rule 5 (with rules 4 and 6, as originally shown) and lastly, one with the 力 character in rule 6 (with rules 4 and 5, as originally shown) as in the original paper (Galley *et al.* 2004). In total, ten different rules are extracted from this training example.

Translation rules are extracted and counted over the entire bilingual training corpus, a count of one for each time they appear in a training example. These counts are used to estimate several features, including maximum likelihood rule probability features such as $P(\text{rule}|e_{\text{root}})$, $P(e_{\text{words}}|f_{\text{words}})$ and $P(f_{\text{words}}|e_{\text{words}})$. We smooth the rule counts with Good-Turing smoothing (Good 1953).

With these models, we can decode a new Chinese sentence by enumerating and scoring all of the English trees that can be derived from it by applying the rules to the Chinese string. The final score of each translation is a weighted product of $P(e)$, the rule probabilities and several other models. One model rewards longer translations, offsetting the language model's desire for short output. Other models punish rules that drop Chinese content words or introduce spurious English content words. A handful of binary features indicate special-purpose rules for translating names/numbers/dates, handling unknown words, copying source-side ASCII text into the translation, or gluing together unparseable or especially long sentences. We also include lexical smoothing models (Gale and Sampson 1996; Good 1953) to help distinguish good low-count rules from bad low-count rules. We optimize the relative weights for these models through minimum error-rate training (Och 2003).

To search efficiently, we employ the CKY dynamic programming parsing algorithm (Yamada and Knight 2002; Galley *et al.* 2006). This algorithm builds English trees on top of Chinese spans. In each cell of the CKY matrix, we store the non-terminal symbol at the root of the English tree being built up. We also store English words that appear at the left and right corners of the tree, as these are needed for computing the $P(e)$ score when

cells are combined. For CKY to work, all transducer rules must be binarized to contain at most two variables. A more efficient search can be gained if this binarization produces rules that can be incrementally scored by the language model (Melamed *et al.* 2004; Zhang *et al.* 2006). Finally, we employ cube pruning (Chiang 2007) for further efficiency in the search.

2.4.4.2.1 Phrase-Based Extraction

The Alignment Template system (ATS) described by Och and Ney (2004) is representative of statistical phrase-based models. The basic unit of translation is the phrase pair, which consists of a sequence of words in the source language, a sequence of words in the target language and a vector of feature values which describe this pair's likelihood. Decoding produces a string in the target language, in order, from beginning to end. During decoding, features from each phrase pair are combined with other features (e.g., reordering, language models) using a log-linear model to compute the score of the entire translation.

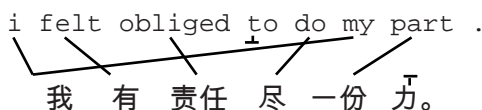


Figure 2.19: a phrase-based training example

The ATS phrase extraction algorithm learns these phrase pairs from an aligned, parallel corpus. This corpus is conceptually a list of tuples of \langle source string, target string, word alignments \rangle which serve as training examples, one of which is shown in Figure 2.19.

For each training example, the algorithm identifies and extracts all pairs of \langle source sequence, target sequence \rangle that are consistent with the alignments. It does this by first enumerating all source-side word sequences up to a length limit L and for each source sequence, it identifies all target words aligned to those source words. For example, in Figure 2.19, for the source phrase 有责任尽, the target words it aligns to are felt, obliged and do. These words and all those between them, are the proposed target phrase. If no words in the proposed target phrase align to words outside of the source phrase, then this phrase pair is extracted.

The extraction algorithm can also look to the left and right of the proposed target phrase for neighboring unaligned words and extracts phrases. For example, for the phrase pair 有责任 \leftrightarrow felt obliged, the word to is a neighboring unaligned word. It constructs new target phrases by adding on consecutive unaligned words in both directions and extracts those in new pairs, too (e.g., 有责任 \leftrightarrow felt obliged to). For efficiency reasons, implementations often skip this step.

Figure 2.20 shows the complete set of phrase pairs up to length 4 that are extracted from the Figure 2.19 training example. Notice that no extracted phrase pair contains the character 我. Because of the alignments, the smallest legal phrase pair, 我有责任尽 \leftrightarrow i felt obliged to do my, is beyond the size limit of 4, so it is not extracted in this example.

有	↔	felt
有责任	↔	felt obliged
有责任尽	↔	felt obliged to do
责任	↔	obliged
责任尽	↔	obliged to do
尽	↔	do
一份	↔	part
一份力	↔	part
一份力.	↔	part .
力.	↔	.
.	↔	.

Figure 2.20: phrases up to length 4 extracted from the example in Figure 2.19

As with GHKM, phrase pairs are extracted over the entire training corpus. Due to differing alignments, some phrase pairs that cannot be learned from one example may be learned from another. These pairs are then counted, once for each time they are seen in a training example and these counts are used as the basis for maximum likelihood probability features, such as $P(f|e)$ and $P(e|f)$.

2.4.4.2.2 Differences in Phrasal Coverage

	Chinese	Arabic
Document IDs	LDC2003E07 LDC2003E14 LDC2005T06	LDC2004T17 LDC2004T18 LDC2005E46
# of segments	329,031	140,511
# of words in foreign corpus	7,520,779	3,147,420
# of words in English corpus	9,864,294	4,067,454

Table 2.34: parallel corpora used to train both models

Both the ATS model and the GHKM model extract linguistic knowledge from parallel corpora, but each has fundamentally different constraints and assumptions. To compare the models empirically, we extracted phrase pairs (for the ATS model) and translation rules (for the GHKM model) from the same parallel training corpora, described in Table 2.34. The ATS model was limited to phrases of length 10 on the source side and length 20 on the target side. A superset of the parallel data was word aligned by GIZA union (Och and Ney 2003) and EMD (Fraser and Marcu 2006a). The English side of training data was parsed using an implementation of Collins' model 2 (Collins 2003).

Table 2.35 shows the total number of GHKM rules extracted and a breakdown of the different kinds of rules. Non-lexical rules are those whose source side is composed entirely of variables: there are no source words in them. Because of this, they potentially apply to any sentence. Lexical rules (their counterpart) far outnumber non-lexical rules. Of the lexical rules, a rule is considered a *phrasal rule* if its source side and the yield of its target side contain exactly one contiguous phrase each, optionally with one or more

variables on either side of the phrase. Non-phrasal rules include structural rules, reordering rules and non-contiguous phrases. These rules are not easy to directly compare to any phrase pairs from the ATS model, so we do not focus on them here.

Statistic	Chinese	Arabic
total translation rules	2,487,110	662,037
non-lexical rules	110,066	15,812
lexical rules	2,377,044	646,225
phrasal rules	1,069,233	406,020
distinct GHKM-derived phrase pairs	919,234	352,783
Distinct corpus-specific GHKM-derived phrase pairs	203,809	75,807

Table 2.35: a breakdown of how many rules the GHKM extraction algorithm produces and how many phrase pairs can be derived from them

Phrasal rules can be directly compared to ATS phrase pairs, the easiest way being to discard the syntactic context and look at the phrases contained in the rules. The second to last line of Table 2.35 shows the number of phrase pairs that can be derived from the above phrasal rules. The number of GHKM-derived phrase pairs is lower than the number of phrasal rules, because some rules represent the same phrasal translation, but with different syntactic contexts. The last line of Table 2.35 shows the subset of phrase pairs that contain source phrases found in our development corpus.

Source of phrase pairs	Chinese	Arabic
GHKM-derived	203,809	75,807
ATS	295,537	133,576
Overlap between models	160,901	75,038
GHKM only	42,908	769
ATS only	134,636	58,538
ATS-useful only	1,994	2,199

Table 2.36: comparison of corpus-specific phrase pairs from each model

Table 2.36 compares these corpus-specific GHKM-derived phrase pairs with the corpus-specific ATS phrase pairs. Note that the number of phrase pairs derived from the GHKM rules is less than the number of phrase pairs extracted by ATS. Moreover, only slightly over half of the phrase pairs extracted by the ATS model are common to both models. The limits and constraints of each model are responsible for this difference in contiguous phrases learned.

GHKM learns some contiguous phrase pairs that the phrase-based extractor does not. Only a small portion of these are due to the fact that the GHKM model has no inherent size limit, while the phrase based system has limits. More numerous are cases where unaligned English words are not added to an ATS phrase pair while GHKM adopts them at a syntactically motivated location, or where a larger rule contains mostly syntactic structure, but happens to have some unaligned words in it. For example, consider Figure 2.21. Because *basic* and *will* are unaligned, ATS will learn no phrase pairs that

translate to these words alone, though they will be learned as a part of larger phrases.

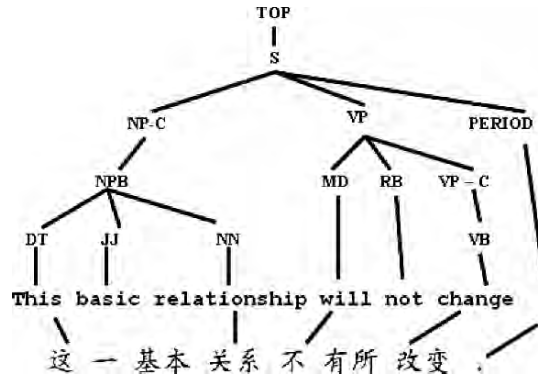
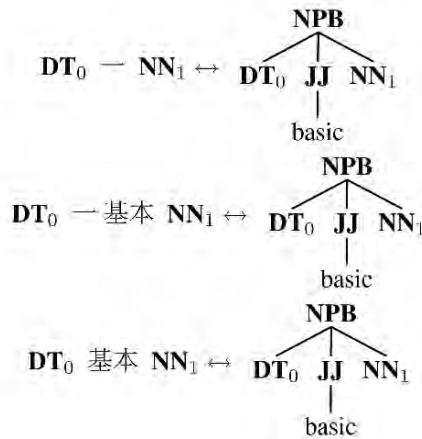


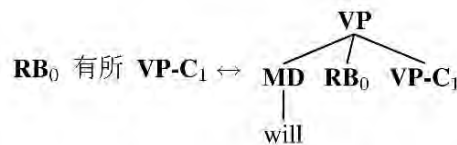
Figure 2.21: Situation where GHKM is able to learn rules that translate into *basic* and *will*, but ATS is not

GHKM, however, will learn several phrasal rules that translate to *basic*, based on the syntactic context

text



and one phrasal rule that translates into *will*



Since we are working with noisy data and noisy alignments, the quality of such additional phrases will vary. For example, the first translation of (literally: “one” or

“a”) to `basic` above is a phrase pair of poor quality, while the other two for `basic` and one for `will` are arguably reasonable.

However, Table 2.36 shows that ATS was able to learn many more phrase pairs that GHKM was not. Even more significant is the subset of these missing phrase pairs that the ATS decoder used in its best translation of the corpus.¹² According to the phrase-based system these are the most “useful” phrase pairs and GHKM could not learn them.¹³ Since this is a clear deficiency, we will focus on analyzing these phrase pairs (which we call *ATS-useful*) and the reasons they were not learned.

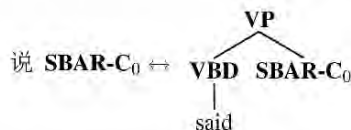
Category of missing ATS-useful phrase pairs	Chinese	Arabic
Not minimal	1,320	1,366
Extra target words in GHKM rules	220	27
Extra source words in GHKM rules	446	799
Other (e.g. parse failures)	8	7
Total missing useful phrase pairs	1,994	2,199

Table 2.37: reasons that ATS-useful phrase pairs could not be extracted by GHKM as phrasal rules

Table 2.37 shows a breakdown, categorizing each of these missing ATS-useful phrase pairs and the reasons they were not able to be learned. The most common reason is straightforward: by extracting only the minimally-sized rules, GHKM is unable to learn many larger phrases that ATS learns. If GHKM can make a word-level analysis, it will do that, at the expense of a phrase-level analysis. Galley *et al.* (2006) propose one solution to this problem and Marcu *et al.* (2006) propose another, both of which are shown in Figure 2.18.

The second reason is that the GHKM model is sometimes forced by its syntactic constraints to include extra words. Sometimes this is only target language words and this is often useful: the rules are learning to insert these words in their proper context. Most of the time, source language words are also forced to be part of the rule and this is harmful: it makes the rules less general. This latter case is often due to poorly aligned target language words (such as the 我 in our Section 2.4.4.2 rule extraction example), or unaligned words under large, flat constituents.

Another factor here: some of the phrase pairs are learned by both systems, but GHKM is more specific about the context of use. This can be both a strength and a weakness. It is a strength when the syntactic context helps the phrase to be used in a syntactically correct way, as in



¹² i.e. highest scoring.

¹³ GHKM may ultimately be able to produce the same translation via a different, possibly lower scoring route.

where the syntax rule requires a constituent of type SBAR-C. Conversely, its weakness is seen when the context is too constrained. For example, ATS can easily learn the phrase

总理 \leftrightarrow prime minister

and is then free to use it in many contexts. But GHKM learns 45 different rules, each that translate this phrase pair in a unique context. Figure 2.22 shows a sampling. Notice that though many variations are present, the decoder is unable to use any of these rules to produce certain noun phrases, such as “current Japanese Prime Minister Shinzo Abe”, because no rule has the proper number of English modifiers.

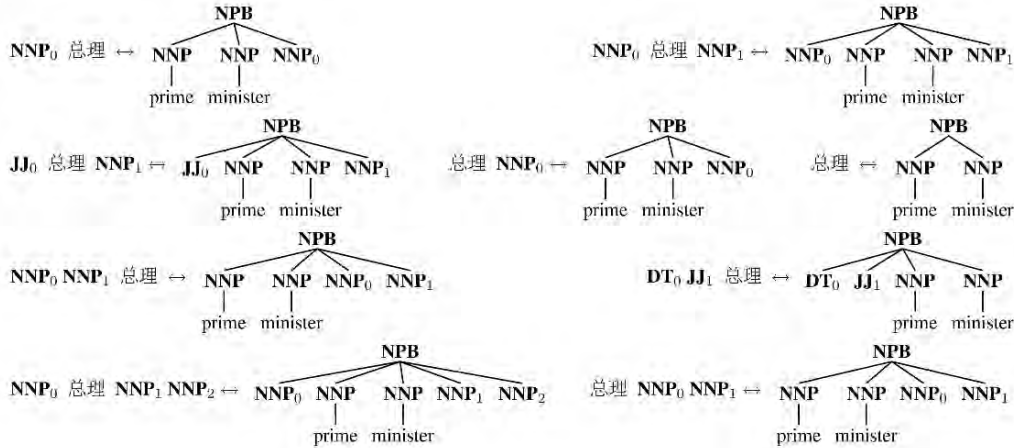


Figure 2.22: a sampling of the 45 rules that translate 总理 to prime minister

2.4.4.2.3 Coverage Improvements

Each of the models presented so far has advantages and disadvantages. In this section, we consider ideas that make up for deficiencies in the GHKM model, drawing our inspiration from the strong points of the ATS model. We then measure the effects of each idea empirically, showing both what is gained and the potential limits of each modification.

Composed Rules. Galley *et al.* (2006) proposed the idea of composed rules. This removes the minimality constraint required earlier: any two or more rules in a parent-child relationship in the derivation tree can be combined to form a larger, composed rule. This change is similar in spirit to the move from word-based to phrase-based MT models, or parsing with a DOP model (Bod *et al.* 2003) rather than a plain PCFG.

Size limit (n)	Chinese	Arabic
0 (minimal)	2,487,110	662,037
2	12,351,297	2,742,513
3	26,917,088	4,824,928
4	55,781,061	8,487,656

Table 2.38: increasing the size limit of composed rules significantly increases the number of rules extracted

Because this results in exponential variations, a size limit is employed: for any two or more rules to be allowed to combine, the size of the resulting rule must be at most n . The size of a rule is defined as the number of non-part-of-speech, non-leaf constituent labels in a rule's target tree. For example, rules 1-5 shown in Figure 2.18 have a size of 0 and rule 6 has a size of 10. Composed rules are extracted in addition to minimal rules, which means that a larger n limit always results in a superset of the rules extracted when a smaller n value is used. When n is set to 0, then only minimal rules are extracted. Table 2.38 shows the growth in the number of rules extracted for several size limits.

In our previous analysis, the main reason that GHKM did not learn translations for ATS-useful phrase pairs was due to its minimal only approach. Table 2.39 shows the effect that composed rule extraction has on the total number of ATS-useful phrases missing. Note that as the allowed size of composed rule increases, we are able to extract a greater percentage of the missing ATS-useful phrase pairs.

Size limit (n)	Chinese	Arabic
0 (minimal)	1,994	2,199
2	1,478	1,528
3	1,096	1,210
4	900	1,041

Table 2.39: Number of ATS-useful phrases still missing when using GHKM composed rule extraction

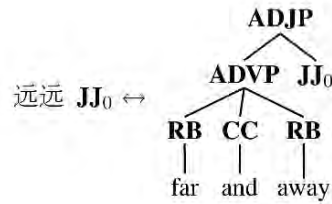
Unfortunately, a comparison of Table 2.38 and Table 2.39 indicates that the number of ATS-useful phrase pairs gained is growing at a much slower rate than the total number of rules. From a practical standpoint, more rules means more processing work and longer decoding times, so there are diminishing returns from continuing to explore larger size limits.

Method	Chinese	Arabic
composed alone (size 4)	900	1,041
SPMT model 1 alone	676	854
composed + SPMT model 1	663	835

Table 2.40: ATS-useful phrases still missing after different non-minimal methods are applied

SPMT Model 1 Rules. An alternative for extracting larger rules called SPMT model 1 is presented by Marcu *et al.* (2006). Though originally presented as a separate model, the method of rule extraction itself builds upon the minimal GHKM method just as composed rules do. For each training example, the method considers all source language phrases up to length L . For each of these phrases, it extracts the smallest possible syntax rule that does not violate the alignments. Table 2.40 shows that this method is able to extract rules that cover useful phrases and can be combined with size 4 composed rules to an even better effect. Since there is some overlap in these methods, when combining the two methods we eliminate any redundant rules.

Note that having more phrasal rules is not the only advantage of composed rules. Here, combining both composed and SPMT model 1 rules, our gain in useful phrases is not very large, but we do gain additional, larger syntax rules. As discussed by Galley *et al.* (2006), composed rules also allow the learning of more context, such as -



This rule is not learned by SPMT model 1, because it is not the smallest rule that can explain the phrase pair, but it is still valuable for its syntactic context.

Restructuring Trees. Table 2.41 updates the causes of missing ATS-useful phrase pairs. Most are now caused by syntactic constraints, thus we need to address these in some way.

Category of ATS-useful phrase pairs	Chinese	Arabic
Too large	12	9
Extra target words in GHKM rules	218	27
Extra source words in GHKM rules	424	792
Other (e.g. parse failures)	9	7
Total missing useful phrase pairs	663	835

Table 2.41: Reasons that ATS-useful phrase pairs are still not extracted as phrasal rules, with composed and SPMT model 1 rules in place

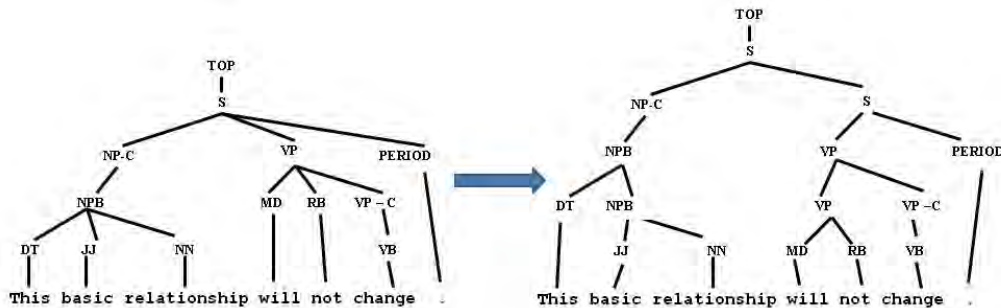


Figure 2.23: Head-out binarization in the target language: S, NPB and VP are binarized according to the head word

GHKM translation rules are affected by large, flat constituents in syntax trees, as in the *prime minister* example earlier. One way to soften this constraint is to binarize the trees, so that wide constituents are broken down into multiple levels of tree structure. The approach we take here is head-out binarization (Wang *et al.* 2007), where any constituent with more than two children is split into partial constituents. The children to the left of the head word are binarized one direction, while the children to the right are binarized the other direction. The top node retains its original label (e.g. NPB), while the new partial constituents are labeled with a bar (e.g. $\overline{\text{NPB}}$). Figure 2.23 shows an example.

Table 2.42 shows the effect of binarization on phrasal coverage, using both composed and SPMT rules. By eliminating some of the syntactic constraints we allow more freedom, which allows increased phrasal coverage, but generates more rules.

Category of missing ATS-useful phrase pairs	Chinese	Arabic
Too large	16	12
Extra target words in GHKM rules	123	12
Extra source words in GHKM rules	307	591
Other (e.g. parse failures)	12	7
Total missing useful phrase pairs	458	622

Table 2.42: Reasons that ATS-useful phrase pairs still could not be extracted as phrasal rules after binarization

2.4.4.2.4 Effect of Coverage Improvements on Translation

	Dataset	# of lines	
		Chinese	Arabic
Development set	NIST 2002 MT eval (sentences < 47 tokens)	925	696
Test set	NIST 2003 MT eval	919	663

Table 2.43: Development and test corpora

To evaluate translation quality of each of these models and methods, we ran the ATS decoder using its extracted phrase pairs and the syntax-based decoder using all the rule sets mentioned above. Table 2.43 describes the development and test datasets used, along with four references for measuring BLEU.

Experiment	Chinese		Arabic	
	Dev	Test	Dev	Test
Baseline ATS	34.94	32.83	50.46	50.52
Baseline GHKM (minimal only)	38.02	37.67	49.34	49.99
GHKM composed size 2	40.24	39.75	50.76	50.94
GHKM composed size 3	40.95	40.44	51.56	51.48
GHKM composed size 4	41.36	40.69	51.60	51.71
GHKM minimal + SPMT model 1	39.78	39.16	50.17	51.27
GHKM composed + SPMT model 1	42.04	41.07	51.73	51.53
With binarization	42.17	41.26	52.50	51.79

Table 2.44: Evaluation results (reported in case-insensitive NIST BLEU4)

Table 2.44 shows the case-insensitive NIST BLEU 4 scores for both our development and test decodings. The BLEU scores indicate, first of all, that the syntax-based system is much stronger at translating Chinese than Arabic, in comparison to the phrase-based system. We believe this is due to the difference in reordering models. While the phrase-based system's reordering comes primarily through the monolingual language model, the syntax rules themselves contain structural reordering decisions learned from the bilingual corpus. Thus syntax-based translation is better able to model the large amount of long-distance reordering required to translate between Chinese and English. The smaller improvements on Arabic-to-English translation reflect the fact that less large scale reordering is necessary.

The BLEU scores also show that syntax-based translation quality benefits from improved phrasal coverage. In addition, rule composition is not only helpful in comparison to minimal rules, but even in addition to SPMT model 1. This indicates that

the size 4 composed rules of Galley *et al.* (2006) bring more than just improved phrasal coverage.

2.4.4.2.5 Discussion

Both the ATS model for phrase-based machine translation and the GHKM model for syntax-based machine translation are state-of-the-art methods. Each extraction method has strengths and weaknesses as compared to the other, and there are surprising differences in phrasal coverage; neither is merely a superset of the other. We have shown that it is possible to gain insights from the strengths of the phrase-based extraction model to increase both the phrasal coverage and translation accuracy of the syntax-based model.

However, there is still room for improvement in both models. For syntax models, there are still holes in phrasal coverage and other areas need progress, such as decoding efficiency. For phrase-based models, incorporating syntactic knowledge and constraints may lead to improvements as well.

2.4.4.3 Improving Hierarchical MT with Dependency Trees

A dependency tree is an alternative grammatical representation to capture long-distance relations between words. For a given sentence, each word has a parent word which it depends on, except for the *root* word. Figure 2.24 shows an example of a dependency tree. Arrows point from children to their parent. In this example, the word *find* is the root.

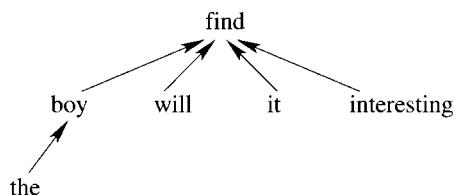


Figure 2.24: The dependency tree for sentence *the boy will find it interesting*

Dependency trees are simpler in form than CFG trees since there are no constituent labels. Furthermore, dependency relations directly model the semantic structure of a sentence. Their simplicity and expressive power make dependency trees a desirable prior model on MT output.

Standard hierarchical phrase-based MT (Chiang 2007) can be viewed as a hierarchical string-to-string model. It does not require explicit syntactic representation on either side of the rules: both the source and target are strings with *NTs*. Decoding are solved as chart parsing.

We extend this model by adding dependency trees to the target side of the model. Our SMT system uses a linear model on a log scale to rank translation hypotheses. To simplify this discussion, we assume a translation model has three dimensions and selects the best hypotheses S'_e for input S_f according to:

$$S'_e = \operatorname{argmax}_{S_e} P(S_e|S_f)^{w_1} P(S_f|S_e)^{w_2} P(S_e)^{w_3}$$

where w_1 , w_2 and w_3 are weights. $P(S_e)$ is the prior probability of the English string S_e using a string language model (LM). In this work we use the terms n -gram LM and string LM interchangeably.

Our decoder uses a dependency LM as the prior. The selecting function becomes

$$D' = \operatorname{argmax}_D P(D|S_f)^{w_1} P(S_f|D)^{w_2} P(D)^{w_3}$$

where $P(D)$ is the dependency LM score of an English dependency tree D .

In practice, we use both a dependency LM and a string LM, as well as several other features in our decoder. Later in this section, we will list the full set of features in our decoder.

2.4.4.3.1 Transfer Rules

A string-to-dependency grammar \mathcal{G} is a 4-tuple $\mathcal{G} = \langle \mathcal{R}, X, T_f, T_e \rangle$, where \mathcal{R} is a set of transfer rules. X is the only non-terminal, similar to the Hiero system (Chiang 2007). T_f is a set of terminals in the source language and T_e is a set of terminals in the target language.¹⁴

A string-to-dependency transfer rule $R \in \mathcal{R}$ is a 4-tuple $R = \langle S_f, S_e, D, A \rangle$, where $S_f \in (T_f \cup \{X\})^+$ is a source string, $S_e \in (T_e \cup \{X\})^+$ is a target string, D represents the dependency structure for S_e and A is the alignment between S_f and S_e . Non-terminal alignments in A must be one-to-one.

In order to exclude undesirable structures and reduce search space, we only allow dependency structure D 's that are either single-rooted (e.g., a complete constituent) or multi-rooted consisting of several sibling dependency trees. We will call such dependency structures *well-formed* in the rest of the work. In addition, the same well-formedness requirement will be applied to partial decoding results. We operate over *well-formed* structures in a bottom-up style in decoding. See the work of Shen *et al.* (2008) for details.¹⁵

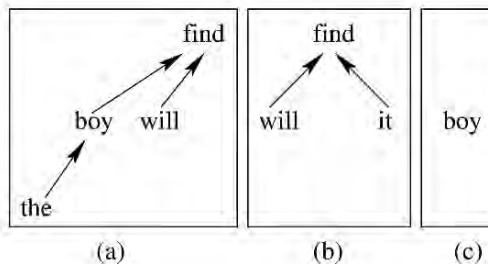


Figure 2.25: single-rooted well-formed structures

¹⁴ We ignore the left hand side here because there is only one non-terminal X . Of course, this formalism can be extended to have multiple NTs.

¹⁵ Single-rooted structures were called *fixed* structures and multi-rooted structures are called *floating* structures by Shen *et al.* (2008).

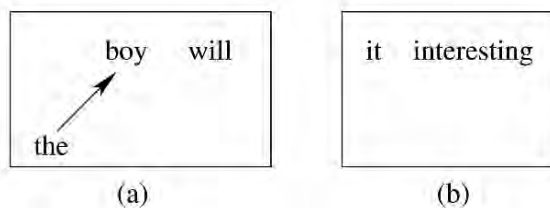


Figure 2.26: multi-rooted well-formed structures

Now we show some examples of single-rooted and multi-rooted *well-formed* structures. We represent dependency structures with graphs. Figure 2.25 shows some examples of single-rooted structures and Figure 2.26 shows some examples of multi-rooted structures.

2.4.4.3.2 Rule Extraction

Now we explain how we get string-to-dependency transfer rules from training data. The procedure is similar to that used by Chiang (2007), except that we maintain tree structures on the target side, instead of strings.

We first use GIZA++ (Och and Ney 2003) to generate forward and backward word level alignment. We take a union of the alignment in both directions. We use a statistical CFG parser to parse the English side of the training data and extract dependency trees with Magerman's rules (Magerman 1995). Then we use heuristic rules to extract transfer rules recursively based on word alignments and target dependency trees. The rule extraction procedure is as follows:

1. Initialization:

All the 4-tuples $(P_f^{i,j}, P_e^{m,n}, D, A)$ are *valid phrase alignments*, where source phrase $P_f^{i,j}$ is aligned to target phrase $P_e^{m,n}$ under alignment A .¹⁶ D is a *well-formed* dependency structure for $P_e^{m,n}$. All *valid phrase templates* are *valid rules templates*.

2. Inference:

Let $(P_f^{i,j}, P_e^{m,n}, D_1, A)$ be a *valid rule template* and $(P_f^{p,q}, P_e^{s,t}, D_2, A)$ a *valid phrase alignment*, where $[p, q] \subset [i, j]$, $[s, t] \subset [m, n]$, D_2 is a sub-structure of D_1 and at least one word in $P_f^{i,j}$ but not in $P_f^{p,q}$ is aligned. We create a new *valid rule template* (P'_f, P'_e, D', A) , where we obtain P'_f by replacing $P_f^{p,q}$ with label X in $P_f^{i,j}$ and obtain P'_e replacing by replacing $P_e^{s,t}$ with X in $P_e^{m,n}$. Furthermore, we obtain D' by replacing sub-structure D_2 with X in D_1 .

¹⁶ By $P_f^{i,j}$ aligned to $P_e^{m,n}$, we mean all words in $P_f^{i,j}$ are either aligned to words in $P_e^{m,n}$ or unaligned and vice versa. Furthermore, at least one word in $P_e^{m,n}$ is aligned to a word in $P_e^{m,n}$.

An example is shown in Figure 2.27.

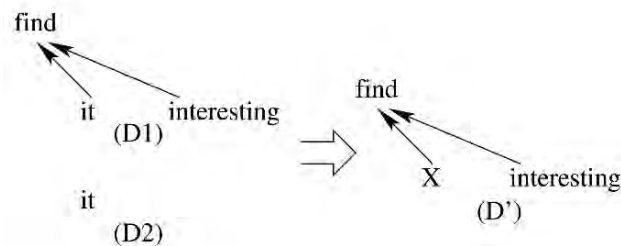


Figure 2.27: Replacing *it* with *X* in D_1

To make the grammar size manageable, each transfer rule can have at most two NTs and at most seven elements (words and NTs) in the source side.

2.4.4.3.3 Decoding

Following previous work on hierarchical MT (Chiang 2005; Galley *et al.* 2006), we decode using as chart parsing. We view the target dependency as a hidden structure of the input string. The task of decoding is then to find the best hidden structure for the input given the transfer grammar and the language models.

The parser scans all source cells in a bottom-up style and checks matched transfer rules according to the source side. Once there is a completed rule, we build a larger dependency structure by substituting component dependency structures for their corresponding NTs in the target side of the rule.

Hypotheses, i.e., candidate dependency structures, are organized in a shared forest, or AND-OR structures. An AND-structure represents an application of a rule over component OR-structures and an OR-structure represents a set of alternative AND-structures with the same *state*.

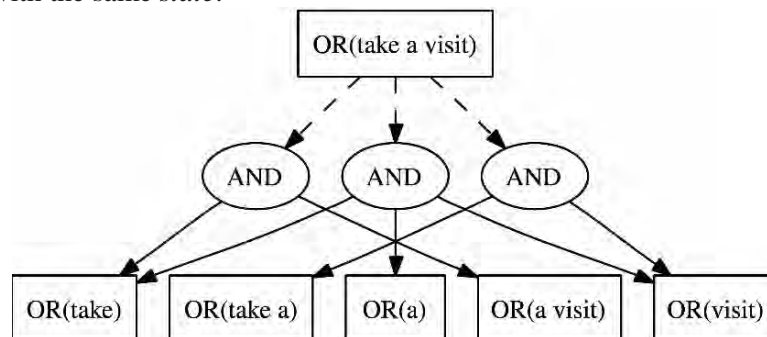


Figure 2.28: An example of AND-OR structures

For example, Figure 2.28 shows part of the AND-OR structures to generate translation *take a visit*. The top OR structure for the whole translation can be built from

one of the three AND structures, which in turn share OR structures as building blocks. The text within the parentheses of the OR structures represent the *states*.

A **state** keeps the necessary information about hypotheses under it, which is needed for computing scores for higher level hypotheses during dynamic programming. For example, with an n -gram string LM in decoding, a state keeps the leftmost $n - 1$ word and rightmost $n - 1$ words shared by hypotheses in that state. Due to the use of a dependency LM in decoding, the state information also includes boundary information about dependency structures for the purpose of computing dependency LM scores for larger structures.

2.4.4.3.4 Dependency Language Model

For the dependency tree in Figure 2.24, we calculate the probability of the tree as follows, which is sometimes called a 3-gram sibling model.

$$\begin{aligned}
 Prob &= P_T(\mathit{find}) \\
 &\times P_L(\mathit{will}|\mathit{find}\text{-as-head}) \\
 &\times P_L(\mathit{boy}|\mathit{will}, \mathit{find}\text{-as-head}) \\
 &\times P_L(\mathit{the}|\mathit{boy}\text{-as-head}) \\
 &\times P_R(\mathit{it}|\mathit{find}\text{-as-head}) \\
 &\times P_R(\mathit{interesting}|\mathit{it}, \mathit{find}\text{-as-head})
 \end{aligned}$$

Here $P_T(x)$ is the probability that word x is the root of a dependency tree. P_L and P_R are left and right side generative probabilities respectively. Let w_h be the head and $w_{L_1}, w_{L_2}, \dots, w_{L_n}$ be the children on the left side from the nearest to the farthest. Suppose we use a 3-gram dependency LM,

$$\begin{aligned}
 P_L(w_{L_1}, w_{L_2}, \dots, w_{L_n} | w_h\text{-as-head}) &= P_L(w_{L_1} | w_h\text{-as-head}) \\
 &\times P_L(w_{L_2} | w_{L_1}, w_h\text{-as-head}) \\
 &\times \dots \times P_L(w_{L_n} | w_{L_{n-1}}, w_{L_{n-2}})
 \end{aligned}$$

In the above formulas, $w_h\text{-as-head}$ represents the event that w is used as the head, while w_{L_i} represents the event that w_{L_i} is a sibling word. This subtle difference is very important, because we need to distinguish a head word from sibling words in context. P_R is similarly computed.

Dependency Decoder Implementation Details

The features in the decoder include:

1. Probability of the source side given the target side of a rule
2. Probability of the target side given the source side of a rule
3. Forward and backward lexical translation probability
4. Number of target words
5. Number of concatenation rules used
6. String LM score

7. Dependency LM score
8. Discount on ill-formed dependency structures

The values of the first four features depend on the rules only and therefore are computed offline. Following Chiang (2005), we also use concatenation rules like $X \rightarrow XX$ for backup. The 5th feature counts the number of concatenation rules used in a translation. In decoding, we also allow dependency structures that are not *well-formed* by treating them as phrasal structures. However, such ill-formed structures are penalized with a discount.

We tune the weights with several rounds of decoding optimization. Following Och (2003), the k-best results are accumulated as the input to the optimizer. Powell's method is used for optimization with 20 random starting points around the weight vector of the last iteration. For improved results, we rescore 1000 best translations, generated using the technique described by Huang and Chiang (2005), by replacing 3-gram string LM scores in the output with 5-gram string LM scores. The algorithm for tuning rescore weights is similar to the one for tuning decoder weights.

2.4.4.3.5 Experiments

We experimented with two translation models:

- baseline: hierarchical string to string translation, using our own replication of the Hiero system (Chiang 2007)
- str-dep: string-to-dependency system. It uses rules with target dependency structures and a dependency LM in decoding.

We use the Hiero model as our baseline because it is the closest to our string-to-dependency model. They use similar rule extraction and decoding algorithms. Both systems use only one non-terminal label in rules. The major difference is that our model represents target translations as dependency structures while Hiero represents them as strings. Thus, the comparison will show the contribution of using dependency information in decoding.

All models were tuned on BLEU (Papineni *et al.* 2001) and evaluated on BLEU, TER (Snover *et al.* 2006) and METEOR (Banerjee and Lavie 2005) so that we could detect over-tuning on one metric. For both Arabic-to-English and Chinese-to-English MT, we tuned on NIST MT02-05 and tested on MT06 and MT08 newswire sets.

The training data for Arabic-to-English MT contains 29M Arabic words and 38M English words. The training data for Chinese-to-English MT contains 107M Chinese words and 132M English words.

The dependency LMs were trained on the same parallel training data. For that purpose, we parsed the English side of the parallel data. Two separate models were trained: one for Arabic from the Arabic training data, and the other for Chinese from the Chinese training data. Traditional 3-gram and 5-gram string LMs were trained on the English side of the parallel data as well as the English Gigaword corpus V3.0 in a way described by Bulyko *et al.* (2007).

Model	Arabic-English	Chinese-English
baseline	337,542,137	193,922,173
str-dep	35,801,341	41,013,346

Table 2.45: Number of transfer rules

Table 2.45 shows the number of transfer rules extracted from the training data. The transfer rules were filtered on the tuning and test sets. The constraint of well-formedness greatly reduced the size of the rule set. The size of the string-to-dependency rule set is about 10% to 20% of the baseline rule set size.

Model	MT06					MT08				
	BLEU		TER		METEOR	BLEU		TER		METEOR
	Lower	mixed	lower	mixed		lower	mixed	lower	mixed	
Decoding (3-gram LM)										
Baseline	47.50	45.48	44.79	46.97	66.17	48.41	46.13	43.83	46.18	67.45
str-dep	48.75	46.74	43.43	45.79	67.18	49.58	47.46	42.80	45.08	68.08
Rescoring (5-gram LM)										
baseline	50.38	48.33	42.64	44.87	67.25	50.50	48.35	42.78	44.92	67.98
str-dep	51.24	49.23	42.08	44.42	67.89	51.23	49.11	42.01	44.15	68.65

Table 2.46: BLEU, TER and METEOR percentage scores on Arabic-English newswire test sets.

Table 2.46 and Table 2.47 show the BLEU, TER and METEOR scores on MT06 and MT08 for Arabic-English and Chinese-English translation respectively.

For Arabic-to-English MT, our string-to-dependency decoder improved BLEU by 1.3 on MT06 and 1.2 and MT08 before 5-gram rescoring. For Chinese-to-English MT, the improvements in BLEU were 1.0 on MT06 and 1.4 on MT08. After rescoring, the improvements became smaller, but still noticeable, ranging from 0.8 to 1.3. TER and METEOR scores were also improved for all conditions, suggesting there was no metric specific over tuning.

Model	MT06					MT08				
	BLEU		TER		METEOR	BLEU		TER		METEOR
	Lower	mixed	lower	mixed		lower	mixed	lower	mixed	
Decoding (3-gram LM)										
Baseline	36.40	34.79	54.98	56.53	57.25	31.64	29.56	57.35	59.37	54.93
str-dep	37.44	35.62	54.64	56.47	57.42	33.05	31.26	56.79	58.69	55.18
Rescoring (5-gram LM)										
baseline	37.88	36.18	53.80	55.45	57.44	33.06	31.21	55.84	57.71	55.18
str-dep	38.91	37.04	53.65	55.45	57.99	34.34	32.32	55.60	57.60	55.91

Table 2.47: BLEU, TER and METEOR percentage scores on Chinese-English newswire test sets.

Table 2.48 shows the interaction between string and dependency LMs. With a tri-gram string LM, a bigram dependency LM is almost as good as a trigram dependency LM. This shows that dependency relations, even in bigram, provide useful prior knowledge unavailable in a larger string n -gram LM. If we use a weaker (bigram) string LM in decoding, the improvement given by a dependency LM is more significant.

		Arabic-English		Chinese-English	
String LM	Dep LM	MT06	MT08	MT06	MT08
2	-	39.92	39.87	29.83	26.35
2	2	43.96	43.93	32.87	28.48
3	-	46.64	47.37	36.02	31.26
3	2	48.55	49.43	37.59	32.64
3	3	48.75	49.58	37.44	33.05

Table 2.48: Lower-cased Bleu scores with various n -gram LMs

2.4.4.3.6 Discussion

Here is an example in the tuning set which may provide some insight how the dependency LM helps to improve translation quality.

- **Reference:** Meriam is a member of the Gulf country Bahrain's royal family
- **Baseline:** Mei-Li-an is the Gulf state of Bahrain's royal family members
- **Str-dep:** Mei-Li-an is a member of the Gulf state of Bahrain's royal family

In the baseline translation, *is ... members* is not a valid combination. However, they are not in a local string window, so the decoder cannot rule it out. However, *members* directly depends on *is* in a dependency structure, so that the decoder uses an alternative rule and obtains a better translation.

2.4.4.3.7 Related Work

The *well-formed* dependency structures defined here are similar to the data structures in previous work on mono-lingual parsing (Eisner and Satta 1999; McDonald *et al.* 2005). Charniak *et al.* (2003) described a two-step string-to-CFG-tree translation model. It first built a target parse forest for the input and then used a syntax-based language model to select the best translation. Unlike our work, the syntax LM was not used in forest generation, possibly due to the complexity of the syntax-based LM. In contrast, the simplicity of dependency LMs makes one step decoding possible in our decoder.

2.4.4.4 Conclusions

This section presents two string-to-tree machine translation models, each using trees to guide large-scale sentence structure changes when translating between languages. One model is based on target-side constituency trees and is improved through a careful comparison with a phrase-based MT model. The other model uses target-side dependency trees to exploit long-distance word relations not captured by traditional n -gram language models, improving on a hierarchical phrase-based MT model. While differences exist in both models, they both succeed by learning a better, more syntactic model to structure the target language output.

2.4.5. Forest Re-ranking for Machine Translation Using the Direct Translation Model

Authors: Zhifei Li and Sanjeev Khudanpur

2.4.5.1 Introduction

Discriminative training has been attempted for the task of statistical machine translation (SMT) both in a *small-scale* setting (i.e., finding optimal weights among a small set of generative models) and in a *large-scale* setting (i.e., training optimal weights for thousands/millions of features). In this section, we focus on the *large-scale* discriminative training.

In the *small-scale* setting, minimum error rate training (Och *et al.* 2003) has become the de facto standard in SMT systems. Smith and Eisner (2006) propose an annealed minimum risk approach, while Zens *et al.* (2007) give a systematic experimental comparison for different training criteria. Shen *et al.* (2004) use perceptron-inspired algorithms to tune weights for tens of features. Chiang *et al.* (2008) use an online max-margin method to tune tens of syntax features for a hierarchical system.

Large-scale discriminative training for SMT is nontrivial due to several reasons. Learning a discriminative model normally involves in running an *iterative* training algorithm, which may require decoding the training data at each iteration. The decoding process is computationally expensive. In particular, the decoding of a single sentence often takes several CPU-seconds and the parallel corpora available for discriminative training typically contain millions of sentence pairs. Therefore, a single decoding pass over the training data may take tens of CPU-days. Another reason that makes discriminative training nontrivial for a SMT task is that the number of features needed to improve MT performance is enormous. For example, the number of phrase pairs extractable from the training *bitext* alone runs into the tens of millions.

To address these problems, previous approaches have resorted either to an *n*-best approximation (e.g., Watanabe *et al.* (2007)), to a computationally cheap baseline system (e.g., a monotone translation system as assumed in Liang *et al.* (2006a)), or to a small-scale setup (e.g., only sentences less than fifteen words are used as by Blunsom *et al.* (2008)).

In this section, we present a *scalable* discriminative re-ranking framework, which discriminatively *re-ranks* hypotheses on a *hyper-graph*, instead on an *n*-best list. Specifically, for each sentence, we generate a hyper-graph using a baseline SMT system and save it to disk. In each iteration of the discriminative training, the learning algorithm updates towards an *oracle tree* in the *fixed* hyper-graph. The re-ranking approach has the advantage of being simple and scalable (since it does not require re-decoding the training data at each iteration of training).

Our hyper-graph-based discriminative re-ranking approach is particularly appealing since a hyper-graph compactly encodes exponentially many hypotheses, representing a much larger hypothesis space than an *n*-best. In this respect, the hyper-graph-based re-ranking occupies an intermediate position between re-ranking on a fixed *n*-best and redecoding the training data at each iteration. To support the hyper-graph-based discriminative training for SMT, we use an *oracle extraction* algorithm that is able to

efficiently extract oracle trees from hyper-graphs and a *hyper-graph pruning* algorithm that substantially reduces the required disk space for saving hyper-graphs without degrading the hyper-graph quality.¹⁷ The scalability of our approach is also due to our simple yet effective *data selection* and *feature selection* algorithms, by which an equally good or better model is obtained using a fraction of the training data and features.¹⁸ Our hyper-graph-based discriminative re-ranking approach is analogous to a lattice-based discriminative training approach for speech recognition (Woodland and Povey, 2002) and a forest re-ranking approach to monolingual parsing (Huang, 2008), both of which have been shown to be quite effective for their respective tasks.

We report experimental results for both the *n*-best and hyper-graph-based discriminative re-ranking on Hiero (Chiang, 2007), showing that our approach is able to improve over a full-scale state-of-the-art hierarchical machine translation system.

2.4.5.2 Hiero and Hyper-graphs

In Hiero (Chiang 2007), a synchronous context-free grammar (SCFG) is extracted from automatically word-aligned corpora. An illustrative grammar rule for Chinese-to-English translation is

$$X \rightarrow \langle X_0 \text{ 的 } X_1, X_1 \text{ of } X_0 \rangle$$

where the Chinese word 的 means of and the alignment, encoded via subscripts on the nonterminals, causes the two phrases around 的 to be reordered around of in the translation. Given a source sentence, Hiero uses a CKY parser to generate a hyper-graph, encoding many hypotheses (derivation trees along with the translation strings).

Formally, a hyper-graph is a pair $\langle V, E \rangle$, where V is a set of nodes (vertices) and E is a set of hyper-edges, with each hyper-edge connecting a set of antecedent nodes to a single consequent node. In parsing parlance, a node corresponds to an item in the chart (which specifies aligned spans of input and output together with a nonterminal label). The root node corresponds to the goal item. A hyper-edge represents an SCFG rule that has been “instantiated” at a particular position, so that the non-terminals on the right and left sides have been replaced by particular antecedent and consequent items; this corresponds to storage of back pointers in the chart.

We write $T(e)$ to denote the set of antecedent nodes of a hyper-edge e . We write $I(v)$ for the set of incoming hyper-edges of node v (i.e., hyper-edges of which v is the consequent), which represent different ways of deriving v .

¹⁷ The details of the oracle extraction algorithm have been presented in Li and Khudanpur (2009).

¹⁸ The details of the data selection method have been discussed in Li and Khudanpur (2008).

Hyper-graphs

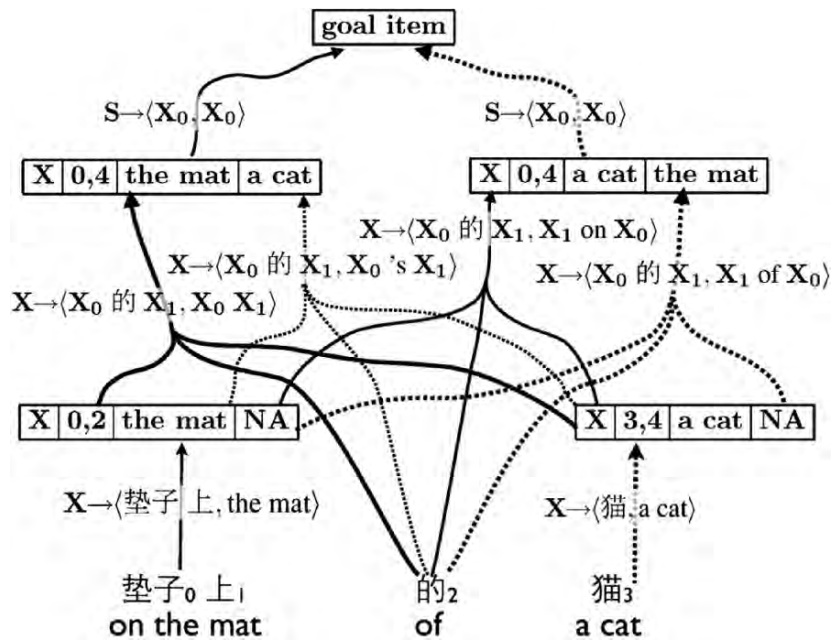


Figure 2.29: A toy hyper-graph in Hiero. When generating the hyper-graph, a trigram language model is integrated. Rectangles represent items, where each item is identified by the non-terminal symbol, source span and left- and right-side language model states. An item has one or more incoming hyper-edges. A hyper-edge consists of a rule and a pointer to an antecedent item for each non-terminal symbol in the rule.

2.4.5.3 Discriminative Modeling

We first describe a general discriminative re-ranking framework, inspired by Roark *et al.* (2007) for n -gram language modeling in speech recognition.

Global Linear Models

A linear discriminant aims to learn a mapping from an input $x \in X$ to an output $y \in Y$, given

1. training examples (x^i, y^i) , $i = 1 \dots N$,
2. a representation $\Phi : X \times Y \rightarrow R^d$ mapping each possible (x, y) to a feature vector,
3. a function $\text{GEN}(x) \subseteq Y$ that enumerates putative labels for each $x \in X$ and
4. a vector $\alpha \in R^d$ of free parameters.

In general, for a SMT task, x is a sentence in the source language; $\text{GEN}(x)$ enumerates possible translations of x into the target language and can be the set of strings in an n -best list or hyper-graph. y is the desired translation: either a reference translation produced by a bilingual human or a so-called oracle translation in $\text{GEN}(x)$ that is most similar to such a human reference. In our training, y is an *oracle tree* (which contains a

translation string as well) since we use the perceptron algorithm, which, unlike the conditional random field used by Blunsom *et al.* (2008), cannot easily treat the derivation trees as latent variables.

Given an input x , the model assigns every hypothesis $y \in \text{GEN}(x)$ a score,

$$\mathbf{s}(x, y) = \Phi(x, y) \cdot \alpha = \sum_j \Phi_j(x, y) \alpha_j, \quad (2.31)$$

where j indexes the feature dimensions.

The **learning** task is to obtain the “optimal” parameter vector from training examples, while the **decoding** task is to search over $\text{GEN}(x)$ to obtain the y that has the maximum score $s(x, y)$. These tasks are discussed in the next sections.

Parameter Estimation

Given a set of training examples, a parameter estimation algorithm is used to find an optimal by maximizing a certain objective function of the training data. Different algorithms use different objective functions, e.g., maximum conditional likelihood (Blunsom *et al.* 2008), minimum risk (Li and Eisner 2009), or max-margin (Chiang *et al.* 2008). We use the averaged perceptron algorithm (Collins 2002) due to its simplicity and suitability to large data settings. Given a set of training examples, the algorithm sequentially iterates over the examples and adjust the parameter vector, as illustrated below:

Perceptron($x, \text{GEN}(x), y$)

1. $\alpha \leftarrow \vec{0}$ ► initialize as zero
2. **for** $t \leftarrow 1$ **to** T
3. **for** $i \leftarrow 1$ **to** N
4. $z^i \leftarrow \operatorname{argmax}_{z \in \text{GEN}(x^i)} \Phi(x^i, z) \cdot \alpha$
5. **if** ($z^i \neq y^i$)
6. $\alpha \leftarrow \alpha + \Phi(x^i, y^i) - \Phi(x^i, z^i)$
7. **return** α

After iterating over the training data a few times, an averaged model, $\frac{1}{T} \sum_{t=1}^T \frac{1}{N} \sum_{i=1}^N \alpha_t^i$, is computed and is used for testing, where α_t^i represents the parameter vector after seeing the i -th example in the t -th iteration, N represents the size of the training set and T is the number of iterations of the perceptron algorithm.

Decoding/Inference

During the perceptron training-time (see line-4 in the perceptron iteration loop) as well as the test time, the following **decision rule** is normally used to select the optimal output y^* ,

$$y^* = \operatorname{argmax}_{y \in \text{GEN}(x)} s(x, y) \quad (2.32)$$

This is called Viterbi decoding. Other decision rules like minimum risk (Tromble *et al.* 2008) or variational decoding (Li *et al.* 2009b) can also be used. The decoding complexity depends on the size and structure of $\text{GEN}(x)$ (e.g., n -best or hyper-graph).

2.4.5.4 Discriminative Forest Re-ranking

Discriminative Re-ranking

The above framework (models, parameter estimation and decoding) is quite general and can be applied in many structured prediction tasks. We use it as a **re-ranking** framework: train a discriminative model to re-rank the hypotheses (encoded either in an n -best list or a hyper-graph) produced by a baseline SMT system. The re-ranking approach has the advantage of being simple and scalable (as we do not need to re-decode the training data following each iteration of training).

Each component $\Phi_j(x, y)$ of the feature vector can be any function of the input x and the output y . To facilitate re-ranking, we first define a baseline feature $\Phi_0(x, y)$, which is the score assigned to y by the baseline SMT system.¹⁹ We then need to define many additional re-ranking features. For example, an n -gram feature might be:

$$\Phi_1(x, y) = \text{Count of the bigram "the of" in } y.$$

Given the feature- and the parameter-vectors, the total score assigned to an output $y \in \text{GEN}(x)$ for a given input x is

$$s(x, y) = \beta \Phi_0(x, y) + \sum_{j \in [1, F]} \alpha_j \Phi_j(x, y), \quad (2.33)$$

where β is the weight for the baseline feature and F is the number of discriminative re-ranking features. To find the optimal weight β for the baseline feature, one could simply treat Φ_0 as a feature in the discriminative re-ranking model and set the value of via the perceptron algorithm. This, however, may lead to undertraining (Sutton *et al.* 2006) of the discriminative re-ranking features in the re-ranking model: the baseline feature is strongly indicative of the overall goodness of y for x , relative to any single discriminative re-ranking feature which indicates the local goodness of y . Therefore, we use a fixed value for during the discriminative training, as suggested by Roark *et al.* (2007).

In the general framework described so far, two algorithms are specific to the size and structure of $\text{GEN}(x)$: *decoding and oracle extraction*. When $\text{GEN}(x)$ is an n -best list as in Li and Khudanpur (2008), these two algorithms are straightforward as we can simply perform a brute-force linear search over the n -best list to obtain the Viterbi or oracle translation. It will be more complex when $\text{GEN}(x)$ is a hyper-graph, which encodes exponentially many hypothesis. We will present the decoding algorithm and review the oracle extraction algorithm (Li and Khudanpur, 2009) in the following sections.

¹⁹ This score itself is often a linear combination of several models, with the relative weights among these models obtained via a minimum error rate training procedure (Och *et al.*, 2003).

Decoding on Hyper-graphs

The decoding algorithm that finds y^* on a hyper-graph, as defined in 2.4.5.2, is outlined below:

Decoding-on-Hyper-graph(GEN(x))

1. **For** v **in** topological order ▶ each node
2. $\hat{s} \leftarrow$ negative infinity
3. **For** $e \in I(v)$ ▶ each incoming hyper-edge
4. $s(e) \leftarrow \beta \Phi_0(e) + \sum_{j \in [1, F]} \alpha_j \Phi_j(e)$
5. $s(e) \leftarrow s(e) + \sum_{j \in T(e)} \hat{s}(u)$
6. **If** $s(e) > \hat{s}(v)$ ▶ better derivation?
7. $\hat{d}(v) \leftarrow \langle e, 1 \rangle$
8. $\hat{s}(v) \leftarrow s(e)$
9. **return** \hat{d} (goal)

Recall that the total score assigned to an output $y \in \text{GEN}(x)$ for a given input x is defined in 2.4.5.3. For each node v , the algorithm adjusts its Viterbi derivation $\hat{d}(v)$ and Viterbi score $\hat{s}(v)$ by applying the discriminative model α . To update the Viterbi derivation $\hat{d}(v)$, the algorithm processes each in-coming hyper-edge $e \in I(v)$. A hyper-edge e 's score $s(e)$ is the model score on that hyper-edge (see line-4 above) plus the Viterbi scores of all its antecedent nodes $u \in T(e)$ (see line-5). If e leads to a better derivation, $\hat{d}(v)$ is updated with $\langle e, 1 \rangle$, the best derivation along the hyper-edge e . The algorithmic complexity is linear with the hyper-graph size, that is, $O(|E|)$.

Oracle Extraction on Hyper-graphs

While a hyper-graph represents a very large set of translations, it is quite possible that the reference translations are not contained in the hyper-graph, due to pruning or inherent deficiency of the translation model. In this case, we want to find the translation in the hyper-graph that is most similar to the reference translations, with similarity computed by BLEU (Papineni *et al.* 2002). Such maximally similar translation will be called *oracle translation* and the process of extracting them *oracle extraction*. As mentioned, we use oracle tree (instead of oracle string) in our training. In Hiero, many distinct derivation trees may yield the oracle string. This is called spurious ambiguity. Among all the trees yielding the oracle string, we define the one with the best baseline model score as the oracle tree.

Oracle extraction on a hyper-graph is a nontrivial task because computing the similarity of any one hypothesis requires information scattered over many items in the hyper-graph and the exponentially large number of hypotheses makes a brute force linear search intractable. Therefore, efficient algorithms that can exploit the structure of the hyper-graph are required. We present an efficient oracle extraction algorithm (Li and Khudanpur 2009), which involves two key ideas. Firstly, we view the oracle extraction as

a bottom-up *model scoring* process on a hyper-graph, where the model is an n -gram *model* “trained” on the reference translation(s) and the *score* is the BLEU value. This algorithm, however, requires maintaining a separate dynamic programming state for each distinguished sequence of “state” words and the number of such sequences can be huge, making the search very slow. Secondly, therefore, we present a novel look-ahead technique, called *equivalent oracle-state maintenance*, to merge multiple states that are equivalent for similarity computation. Our experiments show that the *equivalent oracle-state maintenance* technique significantly speeds up (more than 40 times) the oracle extraction.

2.4.5.5 Supporting Algorithms for Scalable Discriminative Training

In this section, we present several supporting algorithms in the hyper-graph-based discriminative training framework, to make it *scalable* to big training sets and to a large number of features.

Hyper-graph Pruning

Saving the hyper-graphs on the disk requires a lot of storage space and also incurs I/O overheads. Therefore, we adopt a hyper-graph pruning algorithm, which substantially reduces the required disk space without degrading the hyper-graph quality too much. Specifically, we run an *inside-outside* algorithm to compute the Viterbi inside score $\beta(v)$ and the Viterbi outside score $\alpha(v)$ for each node v and then compute the merit $\alpha\beta(e)$ for each hyper-edge e . A hyper-edge e gets pruned if its merit is worse than the score of the best derivation in the hyper-graph by a relative threshold p . A node v gets pruned if all its incoming hyper-edges get pruned. We use a development set to find the optimal p in terms of the tradeoff between disk space and hyper-graph quality (as measured by oracle BLE BLEU). Our algorithm is similar to that used by Huang (2008) for monolingual hyper-graphs.

Feature Selection

The perceptron algorithm previously described itself can be thought as a *feature selection* algorithm as it incrementally adds features into the model whenever they are activated. Due to this, however, the model size can grow arbitrary large, which imposes difficulty in our SMT task as the number of possible features is enormous. For example, the number of phrase pairs extractable from the parallel corpora can easily run into tens of millions. On the other hand, one can employ a dedicated feature selection algorithm and constrain the perceptron algorithm such that it updates only on the selected features.

We propose to use only the features that are present in the *test sets*. Specifically, before the discriminative training, we traverse all the hyper-graphs in the test sets and collect all the features present in these hypergraphs.²⁰ This method is extremely simple and computationally cheap. In machine learning terminology, it is called *directed ad hoc inference*, inference that occupies an intermediate position between *inductive-deductive*

²⁰ Of course, this does not peek at the reference translations.

inference and *transductive* inference (Vapnik 2006). Intuitively, we can view this as an intermediate learning paradigm between the regular supervised training and the unsupervised clustering (e.g., k-nearest neighbors).

Data Selection

We can present all the training examples (i.e., hyper-graphs together with the corresponding oracle trees) to the discriminative training algorithm. Alternatively, we can carry out a data selection procedure that selects a subset of these examples. Data selection is useful for speeding up the training procedure as most training algorithms require computational time proportional to the size of training data. It also has potential to improve the model quality. We adopted the data selection procedure proposed by Li and Khudanpur (2008).

2.4.5.6 Feature Functions

Baseline Model Feature: The baseline feature $\Phi_0(x,y)$ is the score assigned to y by the baseline SMT system, which is a linear combination of several models, with the relative weights among these models obtained via a minimum error rate training procedure (Och *et al.* 2003).

Language Model Features: The *count* of each n -gram in the English yield of the derivation tree y constitutes a feature.

Translation Model Features: The *count* of each *rule* in the derivation tree y constitutes a feature. The rule can be any flat/hierarchical phrase in Hiero.

Boundary n -gram Features: In Hiero, a *hierarchical* rule combines small translation units into a larger one. The combination creates n -grams that cross the boundary between the terminal words in the hierarchical rule and the LM state words in the antecedent items. Some of these boundary n -grams may never appear in monolingual corpora including the English side of the bitext. Clearly, the boundary n -gram features form a *strict subset* of the general n -gram features defined before. For example, all the n -grams inside a *flat* rule and all the unigrams do not belong to the set of boundary n -grams.

2.4.5.7 Experimental Results

We report results using an open source MT toolkit, called Joshua (Li *et al.* 2009a), which implements Hiero (Chiang 2007).

Data Resources

We compile a parallel dataset (about 4M sentence pairs) consisting of corpora distributed by LDC for the NIST Chinese-English MT evaluation, from which we select about 1M sentences pairs (about 28M words in each language) as the parallel training data using a sampling method based on the n -gram matches between training and test sets in the foreign side. The language model training data consists of a 130M words subset of the English Gigaword (LDC2007T07) and the English side of the parallel corpora.

Baselines and Discriminative Training Data

One could train a single baseline SMT system on the entire parallel corpus and decode all the source language sentences using this system, for discriminative training. However, this may lead to a significant mismatch during actual test conditions. In particular, the hyper-graphs generated on the training data will have better translation quality than on unseen test data. To avoid this pitfall, we partition the parallel data into 30 disjoint sections and train 30 baseline SMT systems. To decode sentences from a particular section, we use a baseline system that excludes that section. We carry out this jackknife training for both the TM and LM training.

For each baseline system, we use the GIZA toolkit (Och and Ney 2000), a suffix-array architecture (Lopez 2007), the SRILM toolkit (Stolcke 2002), and minimum error rate training (MERT) (Och *et al.* 2003) to obtain word alignments, translation models, language models and the optimal weights for combining these models, respectively. The baseline LM is a *trigram* LM with modified Kneser-Ney smoothing (Chen and Goodman 1998).²¹ The MERT is performed on the NIST MT'03 evaluation data set.

Goodness of the Oracle Translations

Table 2.49 reports the BLEU scores of the oracle translations on four NIST evaluation data sets on which we will report performance later. This table sets an upper bound on improvements possible using the discriminative model. In general, the oracle BLEU score in a hyper-graph is much better than that in a 500 *unique* n -best strings (corresponding to thousands of distinct derivation trees). This shows that a hyper-graph provides a much better basis for re-ranking than an n -best. Moreover, the oracle BLEU score increases very slowly along with the increase of n for the n -best derivation trees. Huang (2008) observed a similar trend for monolingual parsing.

Task	1-best	Oracle	
		500-unique-best	Hyper-graph
MT'03	35.0	40.7	47.1
MT'04	35.7	44.0	52.8
MT'05	32.6	41.2	51.8
MT'06	28.3	35.1	37.8

Table 2.49: Oracle and 1-best Bleu Scores (4-gram Bleu with four references) on MT Sets.

Goodness of Hyper-graph Pruning

Figure 2.30 shows the tradeoff between disk space (in terms of number of hyper-edges needed to be stored) and hyper-graph quality (in terms of the oracle BLEU score) on a dev set. Clearly, as the threshold p increases, less number of hyper-edges are pruned and the oracle BLEU score decreases less. We choose $p = 7$ for the remaining experiments.

²¹ We use a trigram LM because the decoding of the whole training set (about 1M sentences) is too computationally expensive if we use a larger order LM.

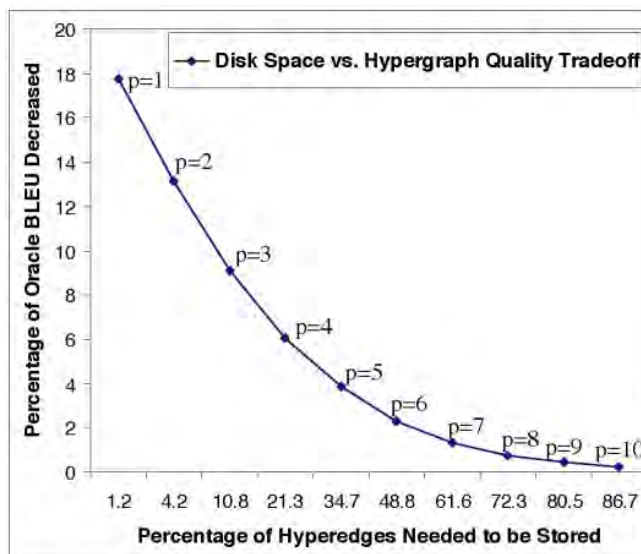


Figure 2.30: Tradeoff between Disk Space and Hyper-graph Quality by varying threshold p .

Results on NIST MT Benchmark Tests

For the experiments reported here, we perform data selection as described in Li and Khudanpur (2008) and select 610K sentence pairs from about 1M for training various perceptron models. The optimal β for the baseline feature is found using MT04 as a tuning set.

Table 2.50 reports the BLEU results for one baseline system, two discriminative n -best re-ranking systems and five discriminative hyper-graph re-ranking systems. BLM means the boundary n -gram language model feature set. In n -best re-ranking, we consider only the regular language model features.²² Clearly, the n -best discriminative training is able to improve the baseline significantly. Also, the “WFS” (with feature selection) model, containing about 160K features, achieves comparable performance to “NFS” (no feature selection) that contains about 1444K features.

System		MT04	MT05	MT06
Baseline	Joshua	35.7	32.6	28.3
N -best	NFS	36.6	33.2	29.3
	WFS	36.5	33.5	29.2
Hyper-graph	LM	35.9	33.0	28.2
	BLM	35.9	33.0	28.4
	TM	36.1	33.2	28.7
	TM+LM	36.0	33.1	28.6
	TM+BLM	36.1	33.0	28.6

Table 2.50: BLEU Scores for MT Sets under Various Models. The MT04 set is used for tuning some hyper-parameters in the re-ranking model.

²² We consider only unigram and bigram for language model features. Adding trigram features does not lead to much improvement. Roark et al. (2007) observed a similar trend for a speech recognition task.

In the hyper-graph-based discriminative re-ranking, we always carry out feature selection. From the test sets (MT03-MT06) we SELECTED 1.5M features for LM and BLM and 1.8M for TM. As is clear in Table 2.50, while the hyper-graph-based discriminative training outperforms the baseline on all three test sets, the improvements in BLEU are slightly less than that from n -best based training. This may be due to overfitting in hyper-graph based training as it yields many more features than n -best based training. Also, all our features so far are “local” and Huang (2008) also observes that forest-based re-ranking in monolingual parsing does not outperform n -best based re-ranking when only local features are used. We propose to investigate these issues in the near future.

2.4.5.8 Conclusions

In this section, we present a *scalable* discriminative training framework for statistical machine translation. The framework can be used both for n -best and *hyper-graph-based* re-ranking. In the context of hyper-graph-based re-ranking, we employ an oracle extraction algorithm to efficiently extract an oracle trees from a hyper-graph. To make the hyper-graph-based method scalable, we adopt several efficient algorithms: hyper-graph pruning, data selection and feature selection. We show that both n -best and hyper-graph-based re-ranking improve over a full scale state-of-the-art hierarchical machine translation system (Chiang 2007). However, the hyper-graph-based re-ranking under performs the n -best re-ranking, which is quite surprising and the reasons remain to be investigated.

Our framework forms a solid basis for feature engineering and incorporating better machine learning methods in statistical machine translation.

2.4.6. Lexical Semantics for Statistical Machine Translation

Authors: Dekai Wu, Pascale Fung, Marine Carpuat, Chi-kiu Lo, Yongsheng Yang, Zhaojun Wu

2.4.6.1 Introduction

Error analysis shows statistical machine translation models to continue to be prone to errors that are likely to benefit from stronger models of semantics. Lexical semantics, in particular, is the obvious first target, given the lexical orientation inherent to SMT approaches. There are two main aspects of lexical semantics — senses and roles — both of which bear exploration in the context of SMT:

Sense disambiguation uses clues in the input context to predict the correct meaning of input lexemes that are ambiguous, thus making or influencing decisions on translation lexical choice.

Semantic role labeling identifies a sentence's shallow semantic predicate-argument frames, independent of syntax or language: who did what to whom, for whom or what, how, where, when and why, which generally should be preserved in translation.

In both these areas of lexical semantics, a large body of work exists, not only in the linguistic tradition, but also, more recently, in extensive empirical and machine learning

oriented evaluations. Yet surprisingly, the application of semantic modeling to SMT has received little or no attention. Leveraging such modeling approaches to improve the adequacy of translation would seem highly desirable in the face of the types of semantic errors made by today's more n -gram based, fluency-oriented SMT systems.

We approach this promise with caution, however, given the painful lessons learned through the historical difficulty of making syntactic models contribute to improving SMT accuracy. The past decade has at last seen increasing amounts of evidence that SMT accuracy can indeed be improved via tree structured and syntactic models (Wu 1997; Wu and Wong 1998; Wu and Chiang 2007; Chiang and Wu 2008; Wu and Chiang 2009) despite numerous disappointing attempts (Och *et al.* 2004). Improving SMT accuracy ultimately required making major adaptations to the original linguistic models. We, therefore, anticipated it would be at least as difficult to successfully adapt even more abstract types of semantic modeling to SMT, and indeed, it has turned out to be remarkably challenging to improve SMT using either kind of lexical semantics, as we will discuss in the respective following sections.

Sense disambiguation: We describe a generalized Phrase Sense Disambiguation approach, the first successful model to incorporate the predictions of a word sense disambiguation system within a typical phrase-based SMT model. The model yields consistent improvement in translation quality across multiple test sets, according not only to BLEU, but to all eight most commonly used automatic evaluation metrics. While preserving the exact structure and feature engineering of state-of-the-art word sense disambiguation (WSD) modeling, success required a simultaneous combination of three other critical adaptations to traditional WSD sense inventories, lexeme definitions and task integration.

Semantic role labeling: We describe results from a series of empirical studies designed to assess more precisely the realistic utility of semantic roles in improving SMT accuracy. The experiments reported study several aspects key to success: (1) the frequencies of types of SMT errors where semantic parsing and role labeling could help, and (2) if and where semantic roles offer more accurate guidance to SMT than merely syntactic annotation, and (3) the potential quantitative impact of realistic semantic role guidance to SMT systems, in terms of BLEU and METOR scores.

2.4.6.2 Semantic SMT: Senses

2.4.6.2.1 Phrase Sense Disambiguation

Phrase Sense Disambiguation (PSD) is a *task-driven* definition of Word Sense Disambiguation (WSD) modeling for SMT. Most SMT systems make little use of contextual information in translation lexical choice. In contrast, WSD research has established that rich context features are useful in stand-alone translation disambiguation tasks. However, early attempts at using context-rich approaches from WSD methods in standard SMT systems surprisingly did not yield the expected improvements in translation quality (Carpuat and Wu 2005).

PSD leverages the feature engineering and learning models developed for standalone WSD, while building on key modeling strengths of current SMT systems. Phrasal SMT architectures directly integrate lexical collocation preferences into lexical choice by the

means of phrasal translation lexicons. The PSD approach argues that disambiguation targets must be exactly the same *phrases* as in the SMT phrasal translation lexicon, so that the sense disambiguation task is identical to lexical choice for SMT.

As a result, PSD contrasts with the conventional standalone WSD perspective. We present comparative empirical evidence and data analysis showing why PSD succeeds where its predecessors failed, via a combination of three critical modeling changes:

1. *The sense disambiguation model must be trained to predict observable senses that are the direct lexical translations of the target lexeme being disambiguated.* PSD sense inventories are exactly the phrasal translations learned in the SMT translation lexicon. In contrast, most conventional WSD models are instead trained to predict hidden senses drawn from an artificially constructed sense inventory. This differs even from previous WSD approaches that make use of word translations as a source of WSD labels, but use manually-defined word-based translation lexicons rather than learned phrasal translations (Dagan and Itai 1994; Gale *et al.* 1992) and still make a distinction between sense labels and SMT translation candidates (Brown *et al.* 1991).

2. *Sense disambiguation must be redefined to move beyond the particular case of single word targets and instead to generalize to multi-word phrase targets.* PSD targets are permitted to be phrasal lexemes composed of smaller lexemes, while standalone WSD targets are typically defined as single words, as in *Senseval* tasks (Kilgarriff and Rosenzweig 1999; Kilgarriff 2001; Mihalcea *et al.* 2004).

3. *The sense disambiguation model must be fully integrated into the runtime decoding.* Unlike earlier models attempting to utilize single-word sense disambiguation (e.g., Carpuat and Wu 2005), it is not possible to represent phrasal sense predictions as input annotations since they cover overlapping spans in the input sentence. PSD is fully integrated into the decoding search itself, as opposed to preprocessing or post-processing stages.

Unlike the static, pre-trained context-independent translation probability models found in nearly all current SMT models, the Phrase Sense Disambiguation approach delivers dynamically-computed, context-dependent, phrasal translation probabilities to the decoder. Whereas, current SMT systems make very little use of contextual information to select a translation candidate for a given input language phrase, PSD employs a rich array of lexical, collocational and syntactic context features from the input source-side sentence, drawing upon the feature engineering of the *Senseval* WSD evaluations. We present analysis confirming that the most important predictive context features are not available in conventional phrase-based SMT models. Further empirical analysis shows that PSD enables the SMT system to make use of more and longer phrases, including phrases that were not seen very frequently in training. Even when the segmentation is identical, the context-dependent lexicons yield translations that match references more often than conventional lexicons.

2.4.6.2.2 PSD uses the same training data as the SMT system

PSD training does not require any other resources than SMT training, or any manual sense annotation. We employ supervised WSD classifiers, since *Senseval* results have

amply demonstrated that supervised models significantly outperform unsupervised approaches (see for instance the English lexical sample tasks results described by Mihalcea *et al.* (2004)).

Training examples are annotated using the phrase alignments learned during SMT training. Every input language phrase is sense tagged with its aligned output language phrase in the parallel corpus. The phrase alignment method used to extract the PSD training data, therefore, depends on the one used by the SMT system. This presents the advantage of training PSD and SMT models on exactly the same data and eliminates domain mismatches between SMT parallel corpora and conventional WSD training data. In addition, this allows PSD training data to be generated entirely automatically, since the parallel corpus is automatically phrase aligned in order to learn the SMT phrasal bi-lexicon.

2.4.6.2.3 PSD generalizes conventional WSD features to phrasal targets

The PSD models are based on the machine learning classifiers and context features used in the conventional WSD models developed for the *Senveval-3* Chinese Lexical Sample Task. However, generalizing the PSD target to multi-word phrases requires generalizing some of the context features as well.

In order to formally define the PSD feature set, consider the input sentence: $w_1 w_2 \dots w_{i-2} w_{i-1} \mathbf{w}_i \dots \mathbf{w}_k w_{k+1} w_{k+2} w_N$ where w_j , for any j , is a single word, p_j is the part of speech of w_j , $\mathbf{w}_i \dots \mathbf{w}_k$ is the PSD phrasal target (a single word if $i=k$ or a multi-word phrase if $k > i$).

The feature set for the PSD phrasal target $\mathbf{w}_i \dots \mathbf{w}_k$ is defined as:

- part of speech of target phrase, which is defined as the POS sequence of its single word components: $p_i \dots p_k$
- part of speech of previous word: p_{i-1} ;
- part of speech of next word: p_{k+1}
- identity of previous word: w_{i-1} ;
- identity of next word: w_{k+1}
- bag of word sentence context: $\{w_j \text{ for } j = 1, \dots, N/p_j = \text{noun, verb or adjective}\}$
- if the target is a single word and a verb, identity of the first noun at the right of the target (approximation for verb-object dependency)
- if the target is a single word and a noun, identity of the next word if it is a noun, i.e., w_{k+1} if p_{k+1} is a noun (approximation for head noun)
- if the target is a single word and a noun, POS of the next word if it is a noun, i.e., p_{k+1} if p_{k+1} is a noun (note that there are several possible POS for nouns)
- if the target is a single word and an adjective, identity of the first noun at the right of the target (approximation for modified noun).

The WSD provenance makes this a significantly richer feature set than previous lexical choice models employed in SMT. Note that some of these features were engineered for Chinese WSD. Shallow parse features can similarly be defined for other specific languages.

2.4.6.2.4 PSD is fully integrated into the SMT decoding

Conventional phrase-based SMT systems are designed to use static phrasal translation lexicons since all translation probabilities and related scores are not dependent of context. In contrast, the PSD context-dependent translation lexicon needs to be dynamically computed as scores for each phrase-pair change for each occurrence of the input language phrase. It is therefore non-trivial to incorporate PSD into an existing phrase-based architecture, since the decoder is not set up to easily accept multiple translation probabilities that are dynamically computed in context-sensitive fashion.

Using PSD scores, therefore, requires making significant changes in the implementation of translation candidate scoring. Note that a generic strategy to overcome this obstacle in any phrasal SMT system is to use a calling architecture that reinitializes the decoder with dynamically generated lexicons on a per-sentence basis.

In practice, in a given input sentence, we may replace each word by an identifier which uniquely represents the occurrence of the word in context. A new translation lexicon is created for these input identifiers using the standard context-independent scores augmented with the new context-dependent PSD score.

2.4.6.2.5 Evaluation

We evaluate the impact of PSD in translation quality within a standard phrase-based SMT architecture.

Experimental setup

The evaluation is conducted on two standard Chinese to English translation tasks. We follow standard machine translation evaluation procedure using automatic evaluation metrics. Since our goal is to evaluate translation quality, we use standard MT evaluation methodology and do not evaluate the accuracy of the WSD model independently.

Preliminary experiments were conducted using training and evaluation data drawn from the multilingual BTEC corpus, which contained sentences used in conversations in the travel domain and their translations in several languages. A subset of this data was made available for the IWSLT06 evaluation campaign (Paul 2006); the training set consists of 40000 sentence pairs and each test set contained around 500 sentences. We used only the pure text data and not the speech transcriptions, so that speech-specific issues would not interfere with our primary goal of understanding the effect of integrating WSD in a full-scale phrase-based model.

A larger scale evaluation was conducted on the standard NIST Chinese-English test set (MT-04), which contained 1788 sentences drawn from newswire corpora and, therefore, of a much wider domain than the IWSLT data set. The training set consisted of about 1 million sentence pairs in the news domain.

Basic preprocessing was applied to the corpus. The English side was simply tokenized and case-normalized. The Chinese side was word segmented using the LDC segmenter.

A standard baseline SMT system

Our aim is to lay out an approach that can be expected to work in any reasonably common phrase-based SMT implementation. Since our focus is not on a specific SMT architecture, we chose the widely-used, off-the-shelf, phrase-based, decoder Moses (Koehn 2004a). Moses implements a beam search decoder for phrase-based statistical models and presents the advantages of being freely available and widely used.

The phrase bi-lexicon was derived from the intersection of bidirectional IBM Model 4 alignments, obtained with GIZA ++ (Och and Ney 2003). A trigram language model with Kneser-Ney smoothing was trained on the English side of the corpus using the SRI language modeling toolkit (Stolcke 2002).

The log-linear model weights were learned using Chiang's implementation of the maximum BLEU training algorithm (Och 2003), both for the baseline and for the WSD-augmented system.

PSD consistently helps translation quality

Table 2.51 shows that, remarkably, PSD consistently improves translation quality as measured by all eight metrics on both the IWSLT and on the NIST data sets. Examples of actual translations are given in Figure 2.31.

Test Set	Exper.	BLEU	NIST	METEOR	METEOR (no syn)	TER	WER	PER	CDER
IWSLT06 Test 1	SMT	42.21	7.888	65.40	63.24	40.45	45.58	37.80	40.09
	SMT+PSD	42.38	7.902	65.73	63.64	39.98	45.30	37.60	39.91
IWSLT06 Test 2	SMT	41.49	8.167	66.25	63.85	40.95	46.42	37.52	40.35
	SMT+PSD	41.97	8.244	66.35	63.86	40.63	46.14	37.25	40.10
IWSLT06 Test 3	SMT	49.91	9.016	73.36	70.70	35.60	40.60	32.30	35.46
	SMT+PSD	51.05	9.142	74.13	71.44	34.68	39.75	31.71	34.58
NIST	SMT	20.41	7.155	60.21	56.15	76.76	88.26	61.71	70.32
	SMT+PSD	20.92	7.468	60.30	56.79	71.34	83.87	57.29	67.38

Table 2.51: results on the IWSLT06 & NIST dataset: integrating the WSD translation predictions improves all metrics across all different available test sets.

PSD integrates useful context in translation decisions

PSD modeling improves lexical choice even in the IWSLT data set. Even in this single domain task, there are genuine sense ambiguities in translation that are addressed by context modeling. Even in a corpus where sentences are only about 10 words long, 18 features per occurrence of a Chinese phrase are observed and used in training PSD models on IWSLT data.

Examining the weights learned by the maximum entropy classifiers for each feature class showed that among the highly useful contextual features are the context-dependent POS and full-sentential bag-of-word context.

<p>IN 没有任何议会投票反对</p> <p>GLOSS <i>not any congressmen vote against he/him</i></p> <p>SMT Without any congressmen voted against him.</p> <p>SMT+PSD No congressmen voted against him.</p>
<p>IN 俄在军臣实行的政策以及对独联体邻国的态度更是令美国担忧。</p> <p>GLOSS <i>Russia in Chechnya implement [de] policy and towards CIS neighbor [de] attitude even more cause US anxiety.</i></p> <p>SMT Russia's policy in Chechnya and CIS neighbors attitude is even more worried that the United States.</p> <p>SMT+PSD Russia's policy in Chechnya an its attitude toward its CIS neighbors cause the United States still more anxiety.</p>
<p>至于美国的人权状况呢?</p> <p>GLOSS <i>as for the U.S. [de] human rights condition?</i></p> <p>SMT As for the U.S. human rights conditions?</p> <p>SMT+PSD As for the human rights situation in the U.S.?</p>
<p>我参拜是为了祈求日本的和平与繁荣</p> <p>GLOSS <i>my visit is for pray for Japan [de] peace and prosperity.</i></p> <p>SMT The purpose of my visit to Japan is pray for peace and prosperity.</p> <p>SMT+PSD The purpose of my visit is to pray for peace and prosperity for Japan.</p>

Figure 2.31: Examples of the impact of PSD modeling on actual translations, drawn from the NIST test set

PSD modeling improves phrasal segmentation for translation

Since automatic evaluation metrics aggregate the effects of very different factors, we also conducted additional data analysis to better understand the contribution of context-dependent PSD modeling (Carpuat and Wu 2008). Interestingly, improvements in translation quality are not only due to better lexical choice for a given target, which could be expected given improvements in metrics such as BLEU, NIST and METEOR, but also due to better phrasal segmentation of the input sentences and better use of the input phrases available in the lexicon. Specifically:

- For the exact same parallel training data, more and longer phrases are used in decoding, including phrases that were seen only few times in training: this suggests that context modeling help better exploit the available vocabulary.
- After compensating for differences in phrasal segmentation, the decoder selects better translations with context-dependent PSD modeling than with conventional context-independent translation probabilities.

These observations are consistent with previous contrastive studies which showed that generalizing to fully phrasal PSD as opposed to conventional single-word WSD is crucial to obtain reliable improvements in translation quality (Carpuat and Wu 2007a).

2.4.6.3 Semantic SMT: Roles

With the encouraging successes in applying sense disambiguation modeling to SMT, we now turn to the other major area of lexical semantics: semantic roles. The most glaring errors made by statistical machine translation systems continue to be those resulting in confusion of semantic roles. These sorts of translation errors often result in serious misunderstandings of the essential meaning of the source utterances — who did what to whom, for whom or what, how, where, when and why.

It has been widely observed that the negative impacts of such errors on the utility of the translation are inadequately reflected by evaluation metrics based on lexical criteria. The accuracy of translation lexical choice has reached increasingly satisfactory levels, at least for largely literal genres, such as newswire, which helps boost lexically oriented scores like BLEU (or METEOR, NIST, or numerous others) despite serious role confusion errors in the translations.

It has also often been noted that precision-oriented metrics such as BLEU tend to reward fluency more than adequacy (in particular, BLEU's length penalty is only an indirect and weak indicator of adequacy). Today's SMT systems produce translations that often contain significant role confusion errors, but nevertheless read quite fluently. Thus, while recent years have seen continued improvement in the accuracy of statistical machine translation systems as measured by such *n*-gram based metrics, this underestimates the effect of the persistent errors of role confusion upon the actual translation utility.

This situation leads us to consider the potential application of shallow semantic parsing and semantic role labeling models to SMT, in ways that might reduce role confusion errors in the translation output. Within the lexical semantics community, increasingly sophisticated models for shallow semantic parsing are being developed. Such semantic parsers, which automatically label the predicates and arguments (roles) of the various semantic frames in a sentence, could automatically identify inconsistent semantic frame and role mappings between the input source sentences and their out-put translations. This approach is supported by the results of Fung *et al.* (2006), which showed that (for the English-Chinese language pairs) approximately 84% of semantic role mappings remained consistent cross-lingually across sentence translations.

However, cautionary lessons come not only again from the difficulty of improving SMT accuracy via syntactic and tree-based models, but also even from the above lexical semantics models for sense disambiguation, which have only finally been successfully applied to increasing SMT accuracy (Carpuat and Wu 2007a; Chan *et al.* 2007; Gimenez and Marquez 2007a) after surprising initial failures (e.g., Carpuat and Wu (2005)). We anticipate at least as much difficulty with semantic role models.

Avoiding the many potential blind alleys calls for careful analysis and evaluation of (1) the frequencies of types of SMT errors where semantic parsing and role labeling could help, (2) if and when semantic roles offer more accurate guidance to SMT than merely syntactic annotation, and (3) the potential quantitative impact of realistic semantic role guidance to SMT systems, at least in terms of scores such as BLEU and METEOR. We now discuss a series of four experiments designed to address each of these questions,

using English-Chinese parallel resources, a typical representative SMT system based on Moses and shallow semantic parsers for both English and Chinese.

2.4.6.3.1 Related work

While this is a new avenue of inquiry, the background relevant to the experiments described here includes: (1) a broad body of work on shallow semantic parsing and semantic role labeling, the majority of which has been performed on English, (2) a relatively small body of work specific to semantic parsing and semantic role labeling of Chinese, and (3) a proposal to measure semantic role overlap as one of the key factors in new MT evaluation metrics.

Shallow semantic parsing

Semantic parsers analyze a sentence with the aim of identifying the “who did what to whom, for whom or what, how, where, when and why.” Shallow semantic parsing extracts the predicate argument structure of verbs in a sentence based on the syntactic tree of that sentence. For example, the predicate argument structure of the verb *hold* in Figure 2.32 specifies a “holding” relation between *both sides* (who) and *meeting* (what) *on Sunday* (when). For a sentence with multiple verbs, there can be multiple predicate argument structures.

Shallow semantic parsing systems are mostly based on classifiers that learn from a manually annotated semantic corpus (Gildea and Jurafsky 2002; Pradhan *et al.* 2005). Following the publication of the Propositional Bank (PropBank) (Palmer *et al.* 2005) first in English, then in Chinese, it has been possible to train these classifiers to perform semantic analysis on news wire type of texts.

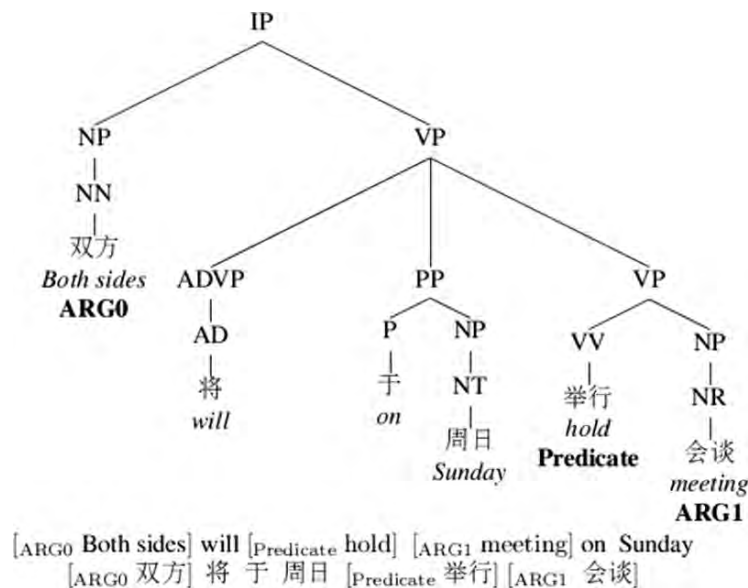


Figure 2.32: Chinese shallow semantic parsing example.

Chinese shallow semantic parsing

Systems that perform shallow semantic parsing on Chinese texts are likewise based on classifiers and trained on the Chinese PropBank and the bilingual English-Chinese Parallel PropBank (Sun and Jurafsky 2004; Xue 2006; Fung *et al.* 2006). It is interesting to note that, despite the very different characteristics of Chinese verbs (Xue and Palmer 2005) from those in English, the core algorithm of a shallow semantic parser remains the same. As was found to be the case in English, SVM classifiers have been found to outperform maximum entropy classifiers for this task (Fung *et al.* 2006). The primary difference lies in the feature set chosen to represent semantic information.

In experiments carried out on PropBank data using gold standard syntactic parse trees, extended syntactic features such as Path Trigram and Path Abbreviations were found to have the highest contribution to system performance (Fung *et al.* 2006). Another feature, Verb Cluster, was also found to be most useful by Xue and Palmer (2005).

MT evaluation metrics based on semantic role overlap

Gimenez and Marquez (2007b, 2008) introduced and refined a set of new MT evaluation metrics employing rich assortments of features reflecting various kinds of similarity at lexical, shallow syntactic, deep syntactic, shallow semantic and deep semantic levels.

Under a number of scenarios, particularly the out-of-domain scenarios, measuring the overlap of shallow semantic roles between the source and target language sentence pairs contributes to improved correlation with human judgment of translation quality. Unsurprisingly, measuring the overlap of manually annotated deep semantic relations contributes even more in some scenarios. However, given the state of automatic semantic parsing technology, realistically, we are today still much closer to being able to incorporate automatic shallow semantic parsing into working SMT systems, and thus, we focus on shallow semantic parsing and semantic role labeling for the present.

2.4.6.3.2 Semantic frames in SMT output

The first of the experiments aims to provide a more concrete understanding of one of the key questions as to the role of semantic parsing in SMT: how well do typical current SMT systems already perform on semantic frames?

The annotated example in Figure 2.33 shows, from bottom to top, (IN) a fragment of a typical Chinese input source sentence that is drawn from newswire text, (REF) the corresponding fragment from its English reference sentence, and (MT) the corresponding fragment of the output sentence from a state-of-the-art SMT system.

A relevant subset of the semantic roles and predicates has been annotated in these fragments. In the Chinese input and its corresponding English reference there are two main verbs marked PRED. The first, (*arrived*), has two arguments: one in an ARG0 agent role, (*Kerry*), and another in an ARG4 destination role, (*Cairo*). The second verb, (*engaged*), has four arguments: one in an ARG0 agent role, again (*Kerry*), one in an

ARG1 role, (*discussions*), and two others in ARGM-MNR manner roles, (*with Mubarak*) and (*on topics*).²³

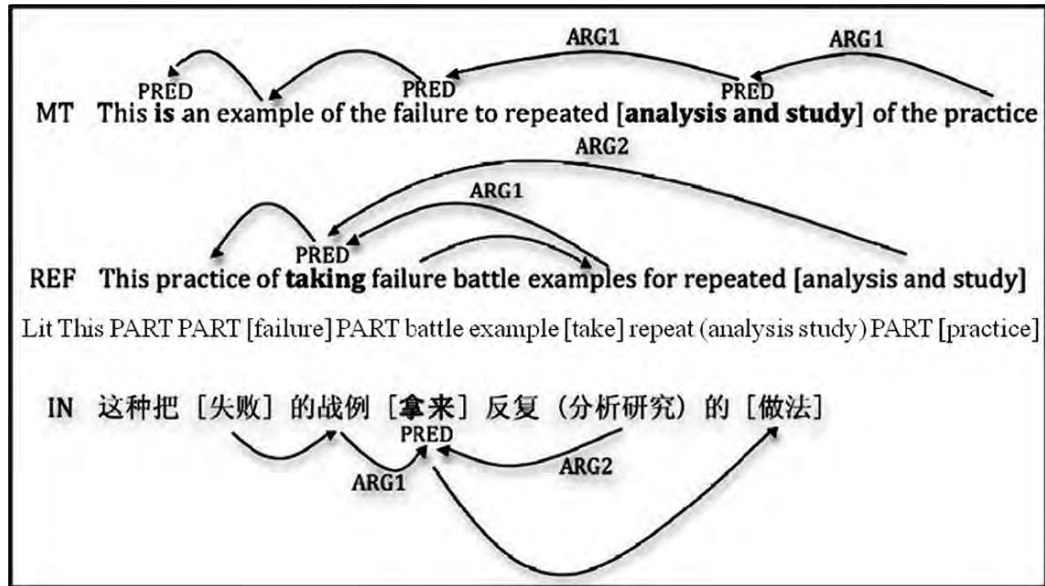


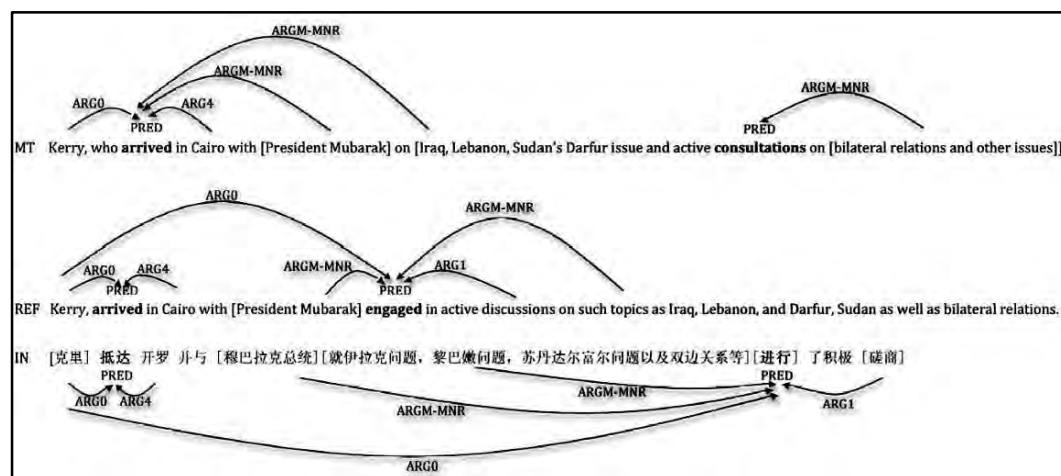
Figure 2.33: Example of semantic frames in Chinese input and English MT output. Lit is the literal translation where PART stands for particle.

In contrast, in the SMT translation output, a very different set of predicates and arguments is seen. While the PRED *arrived* still has the same correct ARG0 *Kerry* and ARG4 *Cairo*, now the ARGM-MNR manner role with *President Mubarak* is incorrectly modifying the *arrived*, instead of an engaged predicate. In fact, the *engaged* predicate has erroneously been completely dropped by the SMT system, so there is no verb to which the arguments of *engaged* can be attached.

Figure 2.34 shows another typical example. Again, PRED marks the main verb in the Chinese input source fragment and its corresponding English reference, (*taking*). It has two arguments: an ARG1 (*battle examples*), and an ARG2 (*analysis and study*).

The SMT translation output, however, not only lacks the main verb, but includes many incorrect predicates and roles. Such spurious predicate-argument structures are clearly seriously detrimental to even cooperative readers straining to guess the meaning of the original Chinese.

²³ Minor variations on the role labeling in these examples are possible, but not central to the present point.



Lit: [Kerry] arrive Cairo with [President Mubarak] [PART Iraq issue Lebanon issue Sudan Darfur issue as well as bilateral relation etc.] [engaged in] PART active [discussion]

Figure 2.34: Example of semantic frames in Chinese input and English MT output.

Experimental setup

To assess the above sorts of phenomena quantitatively, we designed an experiment making use of 745 bi-sentences extracted from the Parallel Prop-Bank with gold standard annotations of both syntactic and semantic roles.

We use the Chinese sentences as system input and their corresponding English translations as the reference translations. We use the open source statistical machine translation decoder Moses (Koehn *et al.* 2007) for the experiments, translating the PropBank Chinese sentences into English with the same model trained for our participation in the IWSLT 2007 evaluation campaign (Shen *et al.* 2007). The English translations generated by the decoder are the system output. Based on the system input and the reference translation, we intend to investigate whether the predicate verbs are correctly translated and their predicate argument structures preserved in the system output.

We first randomly select 50 bi-sentences, without any constraint on the translation accuracy of the predicate verbs, to form the first observation data set (data set A).

Experimental results

Human evaluation of these results show that, for all 138 predicate verbs in the system input (Chinese sentences), only 79 (around 57%) of them are correctly translated in the system output; and given such correctly translated predicate verbs, the translation of their semantic arguments can only achieve around 51% overall F-measure. The detailed results are shown in Table 2.52, data set A.

43% of the Chinese predicate verbs are either not translated at all into English or are translated into a different part-of-speech category such as nouns or adjectives. As shown in Figure 2.35, the predicate verb 位 / *located* in the input Chinese sentence is not translated in the system output.

P-A Structure	Data set A			Data set B		
	Precision	Recall	F-measure	Precision	Recall	F-measure
Predicate	0.98	0.57	0.72	1.00	1.00	1.00
ARG0	0.74	0.38	0.50	0.83	0.66	0.74
ARG1	0.73	0.41	0.53	0.84	0.78	0.81
ARG2	0.82	0.32	0.46	0.80	0.78	0.38
ARG3	1.00	0.67	0.80	0.00	0.00	N/A
ARG4	1.00	0.33	0.51	0.50	1.00	0.66
All ARGs	0.74	0.39	0.51	0.84	0.68	0.75

Table 2.52: Predicate-argument structure translations from Chinese to English in data sets A and B.

<p>IN [ARG0 上述开发区] 基本[PRED 位于][ARG1 福建经济最为活跃的东南部地区]。</p> <p>LIT: [Above mentioned development zone] basically [located] in [Fujian economy most active PART southwestern area].</p> <p>MT: The basic development zones in the southeastern region of Fujian's economy is the most active.</p> <p>REF: The above-mentioned development zones are basically [PRED located] in the Southeastern area of Fujian whose economy is most active.</p>

Figure 2.35: Example of semantic frames in Chinese input and English MT output.

2.4.6.3.3 Semantic roles in SMT output

In the previous experiment, the semantic role accuracy in output translations was negatively affected by errors in identifying the central verb in the first place, as we have seen in both introductory examples of Section 2.4.6.3.2, as well as, the example of Figure 2.32. Without the verb, properly identifying the arguments becomes meaningless. It is, therefore, worth asking a secondary version of the question: *providing the verb is correctly translated*, then how well do typical current SMT systems perform on semantic roles?

Experimental setup

Since nearly half of the predicate verbs in the system input are not translated or wrongly translated in the system output in the previous experiment, we construct another data set (data set B) by randomly selecting 50 bi-sentences under an additional constraint that all predicate verbs are correctly translated. We carry out the same analysis on data set B and the result is shown in the data set B portion of Table 2.52.

Experimental results

For data set B, the overall F-measure of the translation of the semantic arguments is about 75%, which is 24 points higher than that in data set A. In this data set B, we also find that some of the semantic roles are missing in the system output.

<p>IN: 去年四月, C R 公司开始了其破产程序, [ARG0 其所有资产的偿债率]仅[PRED 为] [ARG1 百分之五]。</p> <p>LIT: Last year April, CR company begin PART its bankruptcy procedure, [its all of assets PART compensation rate] only [5 percent].</p> <p>REF: In April of last year, the CR company began bankruptcy procedures and [ARG0 the debt compensation rate of all its assets] [PRED was] only [ARG15%]</p> <p>MT: In April of last year, the company began bankruptcy procedures, all of its debt rate [PRED was] only [ARG1 five percent][ARG0 of the assets] of the CR.</p>

Figure 2.36: Example of semantic frames in Chinese input and English MT output

<p>IN: [ARG0 缅甸和泰国政府] 今天下午在此间[PRED 签订]了 [ARG1 两国边境贸易协定]。</p> <p>LIT: [Burma and Thailand government] today afternoon at here [sign] PART [two country border trade agreement].</p> <p>REF: This afternoon [ARG0 the Myanmaran and Thai governments] [PRE signed] [ARG1 an agreement on border trade between their two countries] here.</p> <p>MT:[ARG? Myanmar and Thailand's government] of [ARG? The two countries border trade</p>

Figure 2.37: Example of semantic frames in Chinese input and English MT output.

A common type of translation error occurs when a group of words that together have a single semantic role in the source language (Chinese) are split into separate groups in the translation (English) often in the wrong word order. In the example of Figure 2.36, the phrase 其所有资产的偿债率 (Its all of capital PART compensation rate) in the input is translated into two separate phrases in the output: *its debt rate* and *of the assets*, creating different semantic relationships compared to the original semantic role of the source phrase. Finally, even though all the words in the arguments of a certain predicate verb are correctly translated into English, their semantic roles are found to be confusing in the translation leading to ambiguity in the interpretation of the translated sentences. As shown in the example of Figure 2.37, although words in both ARG0 and ARG1 are correctly translated into English, we still cannot understand the final translated sentence, because the semantic roles of these two phrases are confused. We cannot tell which semantic roles *Myanmar and Thailand's government* and *the two countries border trade agreements* are supposed to play. This confusion arises from the incorrect position of the predicate verb *signed*.

As we can see, the types of translation errors shown in the examples of Figure 2.35 to Figure 2.37 lead to ambiguity in the final understanding of the translation even though the system output still reads fluently. This is caused by the fact that current *n*-gram based SMT systems are not designed to take semantic roles into consideration.

2.4.6.3.4 Semantic vs. syntactic roles

The third experiment aims to answer another key question: if we favor semantic role consistency across both the source input sentence and the output translation, would this

outperform merely favoring syntactic role consistency across the bi-sentence? In other words, does incorporating semantic role analysis contribute anything beyond the current work on syntactic SMT models?

Experimental setup

To address this question, we perform a different analysis of the previously described set of 745 bilingual sentence pairs with manually annotated syntactic and semantic roles from the Parallel PropBank.

The syntactic roles are manually annotated according to Treebank guidelines. Whereas, the Chinese sentences are annotated with both “subject” and “object” syntactic roles, their English counterparts are only annotated with “subject” roles without the “object” roles.

Furthermore, we manually align the predicate argument structures across the bi-sentences for our experiment. The experiment is done as follows:

1. We first extract all predicate argument structure mappings from the manually annotated and structurally aligned corpus. We compute the statistics of direct semantic role mappings (ARG_i to ARG_i) based on the translation.
2. From the output of step 1, we further look at the syntactic roles associated with each bilingual argument mapping. We use the semantic role boundaries from the annotated corpus to find the syntactic roles.
3. The corresponding Chinese/English syntactic roles are then constructed as syntactic role mappings.

Experimental results

Given all the direct semantic role mappings from Chinese to English, the proportion of syntactic role mappings for the syntactic subject is listed in Table 2.53. We can see that only 84.26% of direct semantic role mappings result from direct syntactic role projections. More than 15% of the subjects are not translated into subjects, even though their semantic roles are preserved across language.

This result shows that semantic roles enforce cross lingual translation patterns than can be done purely by projecting syntactic roles. Whereas syntactic roles vary for each language, semantic roles that convey the meaning of a sentence are translingual.

Syntactic role mapping	Freq	Pct
ZH subject ↔ EN subject	514	84.26%
ZH subject ↔ EN NP	44	7.21%
ZH subject ↔ EN PP	31	5.08%
ZH subject ↔ EN S	15	2.46%
ZH subject ↔ EN other	6	0.98%

Table 2.53: Among gold standard Chinese (ZH) to English (EN) semantic role argument translations, the degree of correspondence of syntactic role mappings.

2.4.6.3.5 Improving SMT with semantic frames

In the fourth experiment, we aim to assess the potential quantitative impact of realistic semantic role guidance to SMT systems, in terms of BLEU and METEOR scores. This is done by simulating the effect of enforcing consistency between the semantic predicates and arguments across both the input source sentence and the translation output.

Experimental setup

For this experiment, we return to data set B, as described in Section 2.4.6.3.3. For each sentence, two types of semantic parse based corrections are permitted to the output translation.

First, the constituent phrases corresponding to either the predicates or the arguments for any labeled semantic role are permitted to be reordered such that a semantic parse of the reordered translation consistently matches the role label on the corresponding phrase in the input source sentence.

Second, if the translation of a predicate in the input source sentence is missing in the output translation, then a translation of that predicate may be added to the output translation such that, again, a semantic parse of the translation consistently associates it with the corresponding arguments for that predicate.

Experimental results

The results, as shown in Table 2.54, show that favoring semantic frame and role consistency across the source input sentence and the output translation improves BLEU and METEOR scores. The accuracy improves on the order of two points, for both metrics. The example of Figure 2.38 shows how two of the constituent phrases are reordered.

It is worth noting that both BLEU and METEOR are still n -gram based metrics, which are of limited accuracy at evaluating fine-grained semantic distinctions in the translations. We suspect that the enhancement in translation quality would be even more obvious under utility-based MT evaluation strategies; this is one main direction for future research.

Metric	Baseline translation	Enforcing consistent semantic parses
BLEU	34.76	36.62
METEOR	63.5	65.9

Table 2.54: SMT performance improvement with semantic predicate and role consistency constraints.

2.4.6.4 Conclusions

Our experiments indicate that statistical machine translation can indeed benefit from lexical semantics, with respect to both sense disambiguation and semantic roles.

With respect to sense disambiguation, we have presented a new Phrase Sense Disambiguation that is the first model to successfully incorporate WSD predictions into phrase-based SMT. The experiments show that three adaptations from traditional WSD configurations are necessary: (1) PSD sense inventories must be the phrasal translations

<p>IN: 加工贸易在广东外贸发展中占有举足轻重的地位, 同时也是粤港澳台经贸合作的重要内容。</p> <p>LIT: Processing trade in Guangdong foreign trade development PART has decisive PART position, same time also is Guangdong Hong Kong Macao Taiwan economy trade cooperation PART important content.</p> <p>REF: The processing trade occupies a crucial position in the development of foreign economy and trade in Guangdong and at the same time is important content in the economic and trade cooperation between Guangdong, Hong Kong, Macao and Taiwan</p> <p>MT: In the processing trade in Guangdong's foreign trade and economic development in Guangdong, Hong Kong, Macao, Taiwan occupies a decisive position at the same time, it is an important content of the economic and trade cooperation.</p>
--

Figure 2.38: Example of semantic frames in Chinese input and English MT output.

learned in the SMT translation lexicon, rather than traditional WSD senses that come from arbitrary sense inventories, (2) PSD targets must be phrases, rather than traditional WSD targets that are single words, and (3) PSD must be fully integrated into the decoding search itself, as opposed to preprocessing or postprocessing stages.

With respect to semantic roles, we have presented a series of experimental studies that illuminate more precisely the likely contribution of semantic roles in improving statistical machine translation accuracy. The experiments reported studied several aspects key to success: (1) the frequencies of types of SMT errors where semantic parsing and role labeling could help, (2) if and where semantic roles offer more accurate guidance to SMT than merely syntactic annotation, and (3) the potential quantitative impact of realistic semantic role guidance to SMT systems, in terms of BLEU and METEOR scores. All sets of results support the utility of shallow semantic parsing and semantic role labeling for improving certain limited but important aspects of SMT accuracy.

These results suggest that lexical semantics has an important role to play in the future of statistical machine translation. Significant adaptations of existing, traditionally configured models will be necessary, however and this should be a major focus of further research.

2.5 Language Modeling for SMT

2.5.1. Distributed Language Modeling

Authors: Ying Zhang, Stephan Vogel, Ahmad Emami, Kishore Papineni, Jeffrey Sorensen and Jerry Quinn

2.5.1.1 Introduction

Statistical language models have a major role in applications such as Automatic Speech Recognition (ASR) (Bahl *et al.* 1983), Statistical Machine Translation (SMT) (Brown *et al.* 1990) and Information Retrieval (Ponte and Croft 1998).

In conventional n -gram language models, the frequencies of all (or a subset of) the n -grams in a corpus are counted, from which the conditional probabilities $P(w_i | w_{i-n+1}^{i-1})$ of

a word given its history of $n-1$ words are computed. Conventional n -gram models have a large number of parameters which need to be reliably estimated. These models are plagued by the high dimensionality problem; in order to estimate their parameters, a large amount of training data is required. In general, the number of unique n -grams increases very fast as the training data size or the n -gram order increases and this makes it very difficult, if not impossible, to scale up standalone n -gram language models (Goodman 2001). Table 2.55 shows the number of n -gram types and tokens in a 600M word English corpus. Training a 5-gram language model for this data will require estimating probabilities and storing the information for 677,159,302 n -grams. This LM will then require several gigabytes of RAM to be loaded into the memory to be used by the SMT or ASR decoders.

n	Type	Token	n	Type	Token
1	1,607,516	612,028,815	6	387,342,304	562,604,106
2	23,449,845	595,550,700	7	422,068,382	562,612,205
3	105,752,368	579,072,585	8	442,459,893	562,632,078
4	221,359,119	562,595,837	9	455,861,099	562,664,071
5	324,990,454	562,598,343	10	465,694,116	562,718,797

Table 2.55: of n -gram types and tokens in a corpus of 600M words.

There are techniques that can reduce the size of a language model, for example using count cut-offs or employing the entropy-based pruning of (Stolcke, 1998). However these methods have their own limitations and it is not possible to keep the size of language models built on ever increasing data without sacrificing some performance.

During the GALE project, Carnegie Mellon University (CMU) and IBM T.J. Watson Research Center independently developed Distributed Language Modeling systems (DLM) to make use of training data in the scale of billions of words. The key idea DLM is to split the large training corpus into manageable chunks and distribute them among a cluster of computers. Each computer hosts a chunk of the training corpus and provides on the fly services to language model queries. In distributed language models, the probability of a word given its history is calculated on the fly based on statistics estimated (also on the fly) from the distributed training corpus. Also, retaining the corpus in its original (split among many computers) brings about the advantage of being able to compute, on the fly, not just the conditional probabilities, but also other type of statistics (e.g. co-occurrence, skip n -gram probabilities, etc). This is not possible with traditional n -gram models where all the statistics (probabilities) are pre-computed and hence fixed.

The CMU approach and IBM approach have many common features in their underlying research ideas and overall system design. However, being independently developed at two separate institutions, they differ in architecture and implementation and are used in different applications. Both systems use *Suffix Arrays* to store the split corpora. Using suffix array allows for efficient computation of different statistics (e.g., n -grams, co-occurrence counts, etc.) from a given corpus. IBM also developed a *CountTree* data structure that allows for much more compact storage of the corpora, but at the disadvantage of being able only to store n -gram type counts, which are the most common used corpus statistics in language modeling.

While the main purpose of developing distributed language model systems was to make building very large language models possible, there are additional advantages of using the proposed distributed architectures. First, these systems are easily scalable; more data can be incrementally added without the need to change or recompute any of the pre-existing modules, and the model can be easily grown as more data becomes available. Second, each part of the corpus can be dynamically assigned a different weight based on the application. Different clients can simultaneously use the distributed corpora with different weights on the corpora. This is done at runtime without any need to rebuild any part of the system. For example, CMU used this flexibility of the architecture to experiment with an *online language model adaptation*.

The IBM system was used with MT decoders, where all the decoder LM queries were submitted to the distributed system. The system was efficient and responsive enough to keep the decoding times short (at most a few times slower than using traditional standalone LMs) while answering queries from multiple decoders simultaneously. All CMU and IBM's ASR experiments were carried out in the N-best rescoring framework. Both IBM and CMU report significant improvement in BLEU scores for the MT experiments and IBM reports gains from using the distributed LM in ASR experiments.

2.5.1.2 Distributed Architecture

The IBM and CMU architectures are very similar. Figure 2.39 illustrates the architecture of the IBM distributed language model. The data is divided among several machines, which we refer to as the *LM workers*. There is no training data stored on the server machine; its sole job is to facilitate communication between the clients and the workers and to compute the language model probabilities.

The communication between the server and the workers is implemented using the Message Passing Interface (MPI) library (Gropp *et al.* 1999). Communication between the clients and the server is implemented via TCP/IP socket networking. This allows a more flexible architecture where clients can connect to and disconnect from the server at will and independently. Clients usually send queries in large batches so as to minimize the TCP/IP communication overhead.

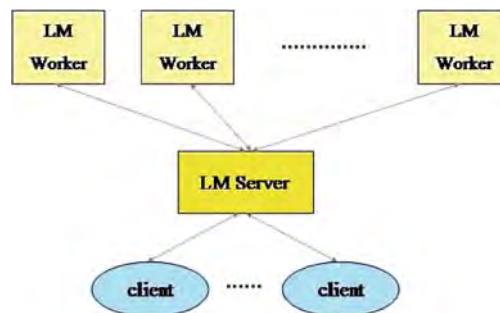


Figure 2.39: Distributed Architecture. Clients can have different weight vectors

Each worker is capable of storing either a whole corpus or a stand-alone, smoothed n -gram language model. Section 2.5.1.3 details the use of suffix arrays for storing a corpus

and for efficient lookups of n -gram counts. A client can request any mixture of the workers, specifying arbitrary weights for each of the component corpora or language models.

2.5.1.3 Suffix arrays

Suffix array was introduced as an efficient method to find instances for a string in a large text corpus. It has been successfully applied in many natural language processing areas (Yamamoto and Church 2001; Ando and Lee 2003). In this section, we will introduce the basic concepts and functions of suffix array.

Indexing the Corpus with Suffix Array

For a monolingual text \mathcal{A} with N words $a_1 a_2 \dots a_N$, we denote the suffix of \mathcal{A} that starts at position i as A_i . A_i is $a_i a_{i+1} \dots a_N$. \mathcal{A} has N suffixes: $\{A_1, A_2, \dots, A_N\}$. Sort these suffixes according to their lexicographical order and we will have a sorted list, such as $[A_{452}, A_{30}, A_1, A_{1511}, \dots, A_7]$. The starting positions of the suffixes in the sorted list form a new array X with N elements, for example, $X = [452, 30, 1, 1511, \dots, 7]$. We call X the *suffix array* of corpus \mathcal{A} .

Formally, suffix array X is a sorted array of the N suffixes of \mathcal{A} , where $X[k]$ is the starting position in \mathcal{A} of the k^{th} smallest suffix. In other words,

$A_{X[1]} < A_{X[2]} < \dots < A_{X[N]}$, where “ $<$ ” denotes the lexicographical order.

The sorting of set $\{A_1, A_2, \dots, A_N\}$ can be done in $\log_2(N+1)$ stages and requires $O(N \log N)$ time in the worst case (Manber and Myers, 1993). The fast sorting algorithm requires additional data structure to keep track of the partially sorted suffixes and thus requires additional memory in indexing. In most of our applications, suffix array only need to be built once and speed is not a major concern. We would rather use a $O(N \log N)$ algorithm which is relatively slower but could index a much larger corpus given the limited amount of memory.

In our implementation, each word a_i is represented by a 4-byte *vocId* and each suffix is a 4-byte pointer pointing to the starting position in \mathcal{A} . Thus $8N$ bytes memory is needed in order to index the corpus \mathcal{A} .

Searching a Corpus Indexed by its Suffix Array

We need to search the suffix array of \mathcal{A} to find all instances of an n -gram \tilde{f} in \mathcal{A} either to measure the frequency of \tilde{f} or to retrieve the actual location of each occurrence.

Definition: For a string u , let $\mathbf{u}_1^p u$ be the prefix consisting of the first p words of u if u contains more than p words and u otherwise. Define the relation $<_p$ to be the lexicographical order of p -word prefixes; that is, $u <_p v$ iff $\mathbf{u}_1^p < \mathbf{v}_1^p$. Relations \leq_p , $=_p$, $>_p$, and \geq_p are defined in a similar way.

Let $l = \min\{k : \tilde{f} \leq_n A_{X[k]} \text{ or } k = N + 1\}$ and $r = \max\{k : A_{X[k]} \leq_n \tilde{f} \text{ or } k = 0\}$. If $l \leq r$, then for any $k \in [l, r]$ both $\tilde{f} \leq_n A_{X[k]}$ and $\tilde{f} \geq_n A_{X[k]}$ are true, meaning that $\forall k \in [l, r], A_{X[k]} =_n \tilde{f}$.

Using a simple binary search in X , we can find l and r using $O(\log N)$ comparisons of strings of size at most n . Each such string comparison requires $O(n)$ single word

comparisons. Thus the suffix array allows us to find all instances of a string \tilde{f} in \mathcal{A} in $O(n \log N)$ time. With l and r found, the number of occurrences of \tilde{f} is $r - l + 1$ and \tilde{f} is located (starting) in \mathcal{A} at positions $X[l], X[l + 1], \dots, X[r]$.

Frequencies of all the substrings in a sentence

In a sentence $\mathbf{f} = f_1, f_2, \dots, f_i, \dots, f_m$, there are m unigram, $m - 1$ bigram, $m - 2$ trigram, ..., and one m -gram in \mathbf{f} . Based on the original algorithm described by Manber and Myers (1990), locating one n -gram in \mathcal{A} requires $O(n \log N)$ in time. For each n -gram, there are total $O(n)$ single-word comparisons for each of the $\log N$ binary searches.

A naive algorithm of locating all substrings in \mathbf{f} thus requires $\log N \cdot \sum_{n=1}^m (m - 2 + 1)n = \frac{m^3 - 3m^2 + 2m}{6} \log N$, which is $O(m^3 \log N)$ in time.

Zhang and Vogel (2005) introduce a faster algorithm for locating frequencies of all substrings in a sentence in $O(m^2 \log N)$ time. It executes m naive searches for the exact substring f_i^m ($i = 1, \dots, m$) and along the trace of the binary search, bookkeeping the suffix array ranges for its prefixes. Each search uses $O(\log N)$ comparisons and each such comparison requires only one word comparison. Thus the search f_i^m is of complexity $O(m \log N)$. The time complexity is $O(m^2 \log N)$ for searching all the substrings in sentence \mathbf{f} .

n	since	2001	after	the	incident	of	the	terrorist
1	2.19×10^4	7559	4.43×10^4	1.67×10^6	2989	6.9×10^5	1.67×10^6	6160
2		165	105	1.19×10^4	1892	34	2.07×10^5	807
3			6	56	106	6	3	162
4				0	0	0	1	0
5					0	0	0	0
6						0	0	0
7							0	0
8								0

attacks	On	the	united	states
9278	2.7×10^5	1.67×10^4	5.1×10^4	3.78×10^4
1398	1656	5.64×10^4	3.72×10^4	3.29×10^4
181	216	545	605	2.58×10^4
35	67	111	239	424
0	15	34	77	232
0	0	10	23	76
0	0	0	7	23
0	0	0	0	7

Figure 2.40: Frequencies of all the embedded n -grams in sentence “since 2001 after the incident of the terrorist attacks on the united states.”

Figure 2.40 shows the frequencies of all the embedded n -grams in sentence “since 2001 after the incident of the terrorist attacks on the united states” matched against a 26 million words corpus. For example, unigram “after” occurs $4.43 \cdot 10^4$ times, trigram “after the incident” occurs 106 times. The longest n -gram that can be matched is 8-gram “of the terrorist attacks on the united states”, which occurs 7 times in the corpus.

Locating the sentence IDs of the n -gram occurrences

The frequency of an n -gram in the training corpus can be used to derive statistics about this continuous n -gram. Sometimes we also want to know the frequency of discontinuous n -grams. A discontinuous n -gram such as “take ... by surprise” is composed from several words or phrases that are not adjacent to each other. These words and phrases have strong dependency with each other and together they form a meaningful unit. One way to estimate the co-occurrences of the discontinuous n -grams is to find out how often all parts of the noncontiguous n -gram occur in the same sentence. To calculate that, we need to know all the sentence IDs for sentences in which n -grams appear.

Given a testing sentence \mathbf{f} of m words, we can use the algorithm described above to find the range $[l,r]$ for an embedded n -gram \tilde{f} of \mathbf{f} . For any $k \in [l,r]$, the suffix $X[k]$ starts with \tilde{f} in the corpus. In other words, we could find all the locations of \tilde{f} in \mathcal{F} as $\{pos \mid X[l], X[l+1], \dots, X[r]\}$.

A naive way to facilitate the conversion from location to sentence ID is to add the sentence ID information in the original corpus before the suffix array indexing. Treat the augmented corpus as a normal text and index it using the suffix array. For a location pos in the corpus, scan towards left of the corpus until a word $\langle s \rangle$ is seen. The next left word contains the sentence ID information and the number of steps taken during scanning is the offset value. For each occurrence, the conversion requires $O(\bar{m})$ steps of scanning, where \bar{m} is the average length of sentences in \mathcal{F} .

To speed up the conversion, we introduce an additional array Δ to record the offset information. $\Delta[pos]$ has the offset value of word a_{pos} with respect to the sentence start of \mathbf{f}_s where a_{pos} lies in. Given a location pos , word a_{pos} is located as the $\Delta[pos]$ -th word in \mathbf{f}_s where $s = a_{pos-\Delta[pos]}$. This operation takes only $O(1)$ in time. Assuming that all the sentences in \mathcal{F} are shorter than 255 words, we only need one byte to represent the offset of each word, which means N additional bytes are needed for the Δ array.

Overall, we use 4 bytes to represent each word, 4 bytes for each suffix pointer and 1 byte for the offset value of each word. In total, we need in total $9N$ bytes to index a corpus of N words.

2.5.1.4 CountTree

Language models constitute that largest static data structures used in our systems, and language model lookups comprise typically 30% or more of our processing time. We designed a new data structure, called the CountTree to facilitate our experiments. This data structure is designed to be space efficient and is organized in a manner that provides efficient access on a single processor, but it is also capable of being partitioned by n -gram prefix in a distributed fashion.

CountTree has the following features:

- Raw 64 bit count representation
- Node compression techniques for reduced memory usage
- Hash tables for fast child node access

- Cache-aware node data layout
- Easily partitionable

By using 64 bit raw counts, different forms of smoothing techniques could be tried while using the same data structure. 64 bit counts are important as the size of data sets head beyond Gigaword corpora.

The basic design of the CountTree LM is a depth-first tree of nodes. The root node is a flat table of arcs to child nodes. Word ids are implicit. Each arc contains a 64 bit unigram count and 64 bit byte offset to the child bigram node.

Basic internal nodes have a 32 bit header storing the number of arcs in the header with several flags. All word IDs are stored in a hash table array following the header. After the word ids, arcs are stored containing a 32 bit count and 32 bit relative address.

To compress a node, the IDs are stored using 16 bits if all are small enough. Counts are also stored in 16 bits if possible. If a child leaf node contains 1 arc, it will be compressed and stored into relative offset of the parent arc. Leaf nodes can implicitly represent count 1 arcs, saving more space.

By storing the word identifiers first, contiguously, the table to be searched is made more compact, reducing the number of CPU cache entries (or cache lines) that must be used during search. Using a cache table prevents the cache misses associated with the probes of a binary search. If an entry is found, only a single data arc needs to be read, using only one additional CPU cache entry.

Because searching is almost always performed in a depth-first manner, it is natural to store the tree data in a similar fashion. Thus, when descending a tree, child nodes are nearby, often in the same memory page. As one moves down the tree, average number of child nodes decreases fast and the search becomes more efficient.

Depth-first storage also makes partitioning easy. Each partition gets one or more subtrees of the LM, starting with bigram nodes. Each sub-tree is contiguous in the LM data-file, simplifying the job of loading only a subset of the data. The root node is stored in the client. Each unigram arc stores a pointer to the appropriate partition in the offset node. When looking up an LM value, the client looks up the root node unigram count, and then sends the n -gram to the partition stored in the offset. When the remaining counts come back, they are combined with the unigram data to compute the final value.

2.5.1.5 Language Modeling

Nativeness of Complete Sentences

We introduce the concept of *nativeness* Q to quantify how likely a sentence \mathbf{e} is generated by a native speaker. A correct sentence should have higher nativeness score than an ill-formatted one. Unlike the sentence *likelihood*, which is defined as the probability of this sentence generated by a language model, nativeness is only a score of real value and is not necessarily probability. Since we cannot ask a native speaker to assign scores to a sentence for its nativeness, we need to estimate this value based on statistics calculated from a collection of sentences \mathcal{E} which resembles what a native speaker would generate.

Before we describe how nativeness is estimated, we first introduce the related notation used in the following discussions. An English sentence \mathbf{e} of length J is a sequence of J words: $e_1, \dots, e_i, \dots, e_j, \dots, e_J$, or e_1^J for short. e_i^j denotes an n -gram $e_i e_{i+1} \dots e_j$ embedded in the sentence. We use \tilde{e} to represent a generic n -gram when n is unspecified. $C(\tilde{e} | \mathcal{E})$ is the frequency of \tilde{e} in corpus \mathcal{E} and $Q(\mathbf{e} | \mathcal{E})$ denotes the nativeness of \mathbf{e} estimated based on \mathcal{E} .²⁴ When there is only one corpus, or the identity of \mathcal{E} is clear from the context, we simply use $C(\tilde{e})$ and $Q(\mathbf{e})$ instead of their full form.

We propose four methods to estimate $Q(\mathbf{e})$ from the data.

- Q_c : Number of n -grams matched.
The simplest metric for sentence nativeness is to count how many n -grams in this sentence can be found in the corpus.

$$Q_c(e_1^J) = \sum_{i=1}^J \sum_{j=1}^J \delta(e_i^j) \quad (2.34)$$

$$\delta(e_i^j) = \begin{cases} 1: C(e_i^j) > 0 \\ 0: C(e_i^j) = 0 \end{cases}$$

For example, Q_c for sentence in Figure 2.40 is 52, because 52 n -grams have non-zero counts.

- $Q_{l(n)}$: Average interpolated n -gram conditional probability.

$$Q_{l(n)}(e_1^J) = \left(\prod_{i=1}^J \sum_{k=1}^n \lambda_k P(e_i | e_{i-k+1}^{i-1}) \right)^{1/J} \quad (2.35)$$

$$P(e_i | e_{i-k+1}^{i-1}) \approx \frac{C(e_{i-k+1}^i)}{C(e_{i-k+1}^{i-1})}$$

$P(e_i | e_{i-k+1}^{i-1})$ is the maximum-likelihood estimation based on the frequency of n -grams. λ_k is the weight for the k -gram conditional probability, $\sum \lambda_k = 1$. λ_k can be optimized by using the held-out data or simply use some heuristics to favor long n -grams.

$Q_{l(n)}$ is similar to the standard n -gram LM except that the probability is averaged over the length of the sentence. This is to prevent shorter sentences being unfairly favored.

- Q_{nc} : Sum of n -grams' non-compositionality. Unlike Q_c where all matched n -grams are equally weighted, Q_{nc} weights each n -gram by their *non-compositionality* (Zhang *et al.* 2006).

²⁴ English is used here for the convenience of description. Most techniques developed in this section are intended to be language-independent.

For each matched n -gram, we consider all possible ways of cutting/decomposing it into two short n -grams, for example “*the terrorist attacks on the united states*” could be decomposed into (“*the*”, “*terrorist attacks on the united states*”) or (“*the terrorist*”, “*attacks on the united states*”), ... , or (“*the terrorist attacks on the united*”, “*states*”). For each cut we calculate the point-wise mutual information (PMI) between the two short n -grams. The one with the minimal PMI is the most “natural” cut for this n -gram. The PMI over the natural cut quantifies the *non-compositionality* I_{nc} of an n -gram w_i^j . The higher the value of $I_{nc}(w_i^j)$ the more likely w_i^j is a meaningful constituent, in other words, it is less likely that w_i^j is composed from two short n -grams just by chance (Yamamoto and Church, 2001). Define Q_{nc} formally as:

$$Q_{nc}(e_1^J) = \sum_{i=1}^J \sum_{j=i+1}^J I_{nc}(e_i^j) \quad (2.36)$$

$$I_{nc}(e_i^j) = \begin{cases} \min_k I(e_i^k; e_{k+1}^j) : C(e_i^j) > 0 \\ 0 & : C(e_i^j) = 0 \end{cases} \quad (2.37)$$

$$I(e_i^k; e_{k+1}^j) = \log \frac{P(e_i^j)}{P(e_i^k)P(e_{k+1}^j)} \quad (2.38)$$

The n -gram probabilities in Equation 2.37 are estimated by the maximum-likelihood estimation.

- Q_t : Sum of pointwise mutual information of the distant n -gram pairs Q_{nc} calculates the PMI of two adjacent n -grams and uses the sum to measure the *non-compositionality* of the sentence. Q_t calculates the PMI of any non-adjacent n -gram pairs based on the co-occurrence of these two n -gram in the same sentences in the training data. The co-occurrences is calculated from the intersection of sentence IDs where each n -gram appear. Sentence IDs are retrieved using methods described in Section 2.5.1.3. We use the sum of all non-adjacent n -gram pairs’ PMI value to measure the *coherence* inside a sentence. This is inspired by the single-word trigger model developed by Rosenfeld (1994). We extend the concept from word-triggers to phrase-triggers in this section. Instead of considering each trigger as a feature and combining all features in a log-linear model, we sum up PMI values of all triggers.

Define Q_t as:

$$Q_t(e_1^J) = \sum_{i_1=1}^{J-g-1} \sum_{j_1=i_1}^{J-g-1} \sum_{i_2=j_1+g+1}^J \sum_{j_2=i_2}^J I(e_{i_1}^{j_1}; e_{i_2}^{j_2}) \quad (2.39)$$

where g is the minimum “gap” length between two n -grams.

Smoothing

n -gram language models require many parameters to be estimated before the model can be used. These parameters include estimates of word string probabilities, as well as, the parameters used for smoothing. Smoothing is one of the most important issues in language modeling. The key idea of smoothing is to discount some probability mass of observed events so that there is a little bit of probability mass left for unseen events. For example, in backoff language models, the weights for backing-off from higher order to lower order n -grams need to be estimated. Similarly, in linearly interpolated models,

$$P_{intp}(w_i|w_{i-n+1}^{i-1}) = \lambda_{w_{i-n+1}^{i-1}} P_{ml}(w_i|w_{i-n+1}^{i-1}) + (1 - \lambda_{w_{i-n+1}^{i-1}}) P_{intp}(w_i|w_{i-n+2}^{i-1}) \quad (2.40)$$

where the maximum likelihood estimate P_{ml} is interpolated with the lower order smoothed model P_{intp} , which is defined in a similar fashion. A model is typically built iteratively starting with the lowest order probabilities (Frederick and Mercer 1980). Estimating a distinct $\lambda_{w_{i-n+1}^{i-1}}$ for each w_{i-n+1}^{i-1} is not practical; in practice the $\lambda_{w_{i-n+1}^{i-1}}$ values are grouped into bins based on the word counts $C(w_{i-n+1}^{i-1})$, where the final weight in each bin is assumed to be equal (Bahl *et al.* 1983).

Normally, in linearly interpolated models, the interpolation weights, $\lambda_{w_{i-n+1}^{i-1}}$ are estimated on some *held-out* data using the EM algorithm (Dempster *et al.* 1977). In a similar fashion, backoff models use *ad-hoc* discounting and backoff parameters that are estimated, or assigned manually, before the language model is used.

CMU Model

Three nativeness metrics proposed in section the beginning of this section ($Q_{l(n)}$, Q_{nc} , and Q_t) are based on the maximum likelihood estimation of n -gram probabilities. $Q_{l(n)}$ interpolates the conditional probabilities using different history length and thus has some smoothing. Other metrics do not have explicit smoothing built in.

Model-best	31.44
$Q_{l(4)}$, equally weighted interpolation	32.16
$Q_{l(6)}$, equally weighted interpolation	32.41
$Q_{l(6)}$, favor long n -grams, $\lambda_k = k/21$	32.40
$Q_{l(6)}$, equally weighted + GT	32.35

Table 2.56: Comparing different smoothing approaches in N -best list re-ranking.

We suspect that smoothing may not be so important for the N -best list re-ranking task as compared to the LM used in the ASR and SMT decoder. If an n -gram is unseen in the training data, assigning 0 as its probability is not acceptable. In a generative probabilistic LM all n -grams have some positive probabilities, no matter how bizarre the n -gram. For us, assigning 0 as the nativeness score to an n -gram is fine because the score is only used to discriminate between two sentences. To test the impact of smoothing on N -best list re-

ranking, we implemented the Good-Turing smoothing (Good 1953) in the $Q_{l(n)}$ metric. Good-Turing (GT) smoothing is easy to implement and the count-of-count information can be obtained on the fly by scanning the suffix-array indexed corpus (Zhang, 2006), which fits our online model set up well. The implementation was validated on a SMT decoder and showed some small improvements over the MLE estimation. However, when applied on the N -best list re-ranking task, GT-smoothing performed slightly worse than using MLE only. Table 2.56 shows the re-ranking results using different smoothing methods. The N -best list is a 10K-list generated by Hiero (Chiang, 2005) on the Chinese MT03 data. Here we used only a 270M word corpus for the re-ranking experiment.

IBM Model

For simplicity, we have decided to use linear interpolation as the smoothing method for our distributed LM architecture. It is also possible to use a backoff smoothing framework, at the cost of a bit more involved lookups and computations. However, we carried out some simple experiments with different smoothing techniques and decided that the choice of smoothing method is less relevant as we move to very large corpora.

Noting that the $\lambda_{w_{i-n+1}^{i-1}}$ are functions of only the counts $C(w_{i-n+1}^{i-1})$, we have instead opted for a closed-form heuristic formula for computing $\lambda_{w_{i-n+1}^{i-1}}$. After plotting the weights $\lambda_{w_{i-n+1}^{i-1}}$ estimated using EM on a held-out set against counts $C(w_{i-n+1}^{i-1})$ we noticed that the plot can be approximately fitted to a *log-linear* curve, from which we estimated the interpolation weights for the distributed language model:

$$\lambda_{w_{i-n+1}^{i-1}} = 0.1 \log_{10} C(w_{i-n+2}^{i-1}) + 0.3 \quad (2.41)$$

from which all the $n + 1$ interpolation weights for the model in Equation 2.40 are computed in an iterative manner starting from the highest order n -gram.

The clear advantage of this heuristic formula is that $\lambda_{w_{i-n+1}^{i-1}}$ is directly computable from word counts in the training data. It should also be noted that there is no lookup overhead here, since the count $C(w_{i-n+2}^{i-1})$ needs to be extracted to estimate the probability $P_{ml}(w_i | w_{i-n+1}^{i-1})$.

Training	λ estimation	Perplexity
I	EM	193
I	heuristic	216
II	EM	128
II	heuristic	137

Table 2.57: Perplexities for different interpolation weight estimation methods

Table 2.57 shows perplexities of linear interpolated 5-gram models using both the bucketed EM estimate and heuristic interpolation weights. There are two different training data sets used, denoted by *I* and *II* in the table and consisting of approximately 200M and 2.9G words respectively. The held-out data used for EM estimation of the

weights is a corpus of 35M words. The vocabulary size is 128K and the perplexities are reported on an unseen test set of 58M words.

As can be seen in the table the linearly interpolated model with heuristic interpolation weights has comparable perplexity to those estimated using the EM algorithm. It should also be noted that EM maximizes the performance of the model for likelihood and hence, the perplexity. We expect the difference between the heuristic and EM estimated weights to be smaller when used for a practical task such as ASR or SMT where the performance is measured by metrics, which are not fully correlated with perplexity.

Smoothed LM workers

It is possible to have LM workers that store an already smoothed language model instead of a corpus. These workers do not provide the server with counts and can only send smoothed probabilities for the queries n -grams. In case the collection of LM workers contains both corpora (suffix arrays) and smoothed LMs; the server must somehow combine all the information from the workers to compute a final smoothed probability. Assume there are m_1 workers with suffix array corpora and m_2 standalone smoothed n -gram models for a total of $k = m_1 + m_2$ workers. Assuming that the j^{th} client is using the worker mixture vector $(s_1^j, s_2^j, \dots, s_k^j)$, then the aggregate count is simply $C^j(W_1^n) = \sum_{i=1}^{m_1} s_i^j C_i(W_1^n)$ where $C_i(W_1^n)$ is the count of n -gram W_1^n for the corpus stored at client i .

In case the client requested probabilities, the aggregate probability $P_a^j(w_n|W_1^{n-1})$ (for workers 1 through m_1) is computed according to Equation 2.40 and Equation 2.41 above, F using the aggregate statistics C . Taking into account the standalone smoothed LM workers, the final probability is defined as follows:

$$P^j(w_n|W_1^{n-1}) = \alpha^j P_a^j(w_n|W_1^{n-1}) + (1 - \alpha^j) \sum_{i=1}^{m_2} \frac{s_i^j}{\sum_{l=1}^{m_2} s_l^j} P_w^i(w_n|W_1^{n-1}) \quad (2.42)$$

where $P_w^i(w_n|W_1^{n-1})$ is the probability from the i^{th} standalone worker and $\alpha^j = \frac{\sum_{i=1}^{m_1} s_i^j}{\sum_{i=1}^{m_1+m_2} s_i^j}$.

2.5.1.6 Experiments and Results

We carried out experiments using our distributed language model for both speech recognition and machine translation tasks. In the machine translation experiments, the distributed language model was used by CMU to re-rank the N -best list and experiment with online language model adaptation. IBM directly integrated the distributed LM into the decoder. In the speech recognition experiments, the distributed model was used only in re-ranking N -best lists created by a speech recognition decoder using standard language models.

CMU N -best List Re-ranking Results

CMU uses the distributed language model on very large data to re-rank the N -best list.

Table 2.58 lists results of the re-ranking experiments under different conditions. The re-ranked translation improved the BLEU score from 31.44 to 32.64, significantly better than the model-best translation.

# of Relevant Chunks per. Sent	1	2	5	10	20	150
offline 3-gram KN	32.22					32.08
offline 4-gram KN	32.22					32.53
Q_c	32.27	32.38	32.40	32.47	32.51	32.48
$Q_{l(3)}$	32.00	32.14	32.14	32.15	32.16	
$Q_{l(4)}$	32.18	32.36	32.28	32.44	32.41	
$Q_{l(5)}$	32.21	32.33	32.35	32.41	32.37	
$Q_{l(6)}$	32.19	32.22	32.37	32.45	32.40	32.41
$Q_{l(7)}$	32.22	32.29	32.37	32.44	32.40	
Q_{nc}	32.29	32.52	32.61	32.55	32.64	32.56

Table 2.58: BLEU scores of the re-ranked translations. Baseline score = 31.44

Different metrics are used under the same data situation for comparison. Q_c , though extremely simple, gives quite nice results on N -best list re-ranking. With only one corpus chunk (the most relevant one) for each source sentence, Q_c improved the BLEU score to 32.27. We suspect that Q_c works well because it is in line with the nature of BLEU score. BLEU measures the similarity between the translation hypothesis and human reference by counting how many n -grams in MT can be found in the references.

Instead of assigning weights 1 to all the matched n -grams in Q_c , Q_{nc} weights each n -gram by its *non-compositionality*. For all data conditions, Q_{nc} consistently gives the best results.

Metric family $Q_{l(n)}$ is close to the standard n -gram LM probability estimation. Because no smoothing is used, $Q_{l(3)}$ performance (32.00) is slightly worse than the standard 3-gram LM result (32.22). On the other hand, increasing the length of the history in $Q_{l(n)}$ generally improves the performance.

System	LM Size	LM Type	BLEU	NIST
Baseline	200 M	3-gram	31.44	9.01
Re-ranked	2900 M	3-gram	32.08	9.14
		4-gram	32.53	9.19
	20M per sent.	Q_{nc}	32.34	9.20
		Q_t	32.46	9.21
		$Q_{nc} + Q_t$	32.60	9.24
	200M per sent.	Q_{nc}	32.60	9.23
Q_t		32.79	9.25	
		$Q_{l(6)} + Q_t$	32.90	9.28

Table 2.59: Re-ranking N -best list using distributed language model.

Table 2.59 compares re-ranking results using standard 3-gram and 4-gram language model with using Q_{nc} , Q_t features. For each testing sentence, most relevant corpus chunks of 20M/200M words are used as the training data to calculate the nativeness scores for re-ranking. From this result, we can see that by considering long distance dependencies of n -grams in a sentence (Q_t) improves the discriminative power of the language model.

CMU Online Language Model Adaptation

Although statistical systems usually improve with more data, performance can decrease if additional data does not resemble the test data. We would prefer to use data that is more “relevant” to the testing data than using all data in a bag.

The concept of “relevance” depends on the application. In Information Retrieval (IR), a document is considered to be “relevant” to a user’s search intention if it contains the query words. For this purpose, features like TF-IDF score which capture the key words of a document work well in estimating the “relevance” and are widely used in IR. For statistical machine translation, however, a relevant corpus chunk is the one on which we can train a better language model that helps the decoder to generate a better translation. The quality of the translation is evaluated by calculating the n -gram precision and recall against human generated reference translations. To be consistent with the evaluation methods, we consider using n -gram based features to predict the relevance of a corpus chunk to a testing sentence.

Relevance measurements

Here we study several simple statistics to select relevant corpus chunks to build LM for each testing sentence \mathbf{f} . These relevance statistics are calculated over the N -best translations of \mathbf{f} .

Perplexity

The standard measurement for language model quality is perplexity. Considering the N -best list as a document of N English sentences, we can calculate the standard language model perplexity of the N -best list using the language model trained on each of the corpus chunk \mathcal{E} . Denote the N -best translation list of \mathbf{f} as T . T contains $|T|$ English words from all N translations. The perplexity of language model trained on \mathcal{E} is measured on $T = e_1 \dots e_{|T|}$:

$$PP(T | \mathcal{E}) = \exp \left(-\frac{1}{|T|} \sum_{i=1}^{|T|} \log p(w_i | w_{i-n+1}^{i-1}) \right) \quad (2.43)$$

The pitfall here is that two perplexities are not directly comparable if two language models are trained from different corpus. Perplexities are only comparable when two corpora have the same vocabulary size which is most likely not true as we only split the corpora into chunks of similar size without considering the resulting vocabulary size.

Bag-of- n -gram Coverage Rate

Define the *bag-of- n -gram coverage rate* of corpora \mathcal{E}_d to sentence \mathbf{f}_t as:

$$R(\mathcal{E}_d; \mathbf{f}_t) = \sum_{r=1}^N Q_c(\mathbf{e}_t^{(r)} | \mathcal{E}_d) \quad (2.44)$$

For hypothesis $\mathbf{e}_t^{(r)}$, the r^{th} hypothesis in the N -best list, $Q_c(\mathbf{e}_t^{(r)} | \mathcal{E}_d)$ is the number of n -gram tokens in the hypothesis that can be found in \mathcal{E}_d (see Eqn. $Q_c(\mathbf{e}_1^j) = \sum_{i=1}^j \sum_{j=1}^j \delta(\mathbf{e}_i^j)$).

Informally, $R(\mathcal{E}_d; \mathbf{f}_t)$ estimates how well \mathcal{E}_d covers n -grams (tokens) in the N -best list of \mathbf{f}_t . The higher the coverage, the more relevant \mathcal{E}_d is to \mathbf{f}_t .

Weighted n -gram Coverage Rate

Unlike the bag-of- n -gram coverage rate where n -grams of different orders are considered equally, weighted n -gram coverage rate applies different weights on n -gram coverage rates of different orders. Intuitively, we should predict a corpus chunk to be more relevant than another if it has a higher high-order n -gram coverage rate.

Define $M(\mathcal{E}, \mathbf{f})$, the *weighted n -gram coverage rate* of corpus \mathcal{E} for a sentence \mathbf{f} as: where $\text{Coverage}_k(\mathcal{E}, \mathbf{f})$ is the k^{th} order n -grams coverage rate of the corpus on \mathbf{f} 's N -best list:

$$M(\mathcal{E}, \mathbf{f}) = \sum_{k=1}^n \lambda_k \text{Coverage}_k(\mathcal{E}, \mathbf{f}) \quad (2.45)$$

where $\text{Coverage}_k(\mathcal{E}, \mathbf{f})$ is the k^{th} order n -grams coverage rate of the corpus on \mathbf{f} 's N -best list:

$$\text{Coverage}_k(\mathcal{E}, \mathbf{f}) = \frac{\sum_{e_i^{i+k-1} \in \{e_t^{(1)} \dots e_t^{(N)}\}} C(e_i^{i+k-1}) \Phi(e_i^{i+k-1}, \mathcal{E})}{\sum_{e_i^{i+k-1} \in \{e_t^{(1)} \dots e_t^{(N)}\}} C(e_i^{i+k-1})} \quad (2.46)$$

$C(e_i^{i+k-1})$ is the frequency of n -gram type e_i^{i+k-1} 's occurrences in the N -best list and

$$\Phi(e_i^{i+k-1}, \mathcal{E}) = \begin{cases} 1, & \text{if } e_i^{i+k-1} \text{ exists in } \mathcal{E} \\ 0, & \text{otherwise.} \end{cases} \quad (2.47)$$

Informally, we predict a corpus chunk \mathcal{E}_d to be most "relevant" to a testing sentence \mathbf{f} if n -grams in \mathbf{f} 's N -best translations can be most covered by \mathcal{E}_d .

Experiments

The experiment is set up as follows. The LDC Gigaword corpus (both version 2 and version 3) of 4,916,610,128 words is split into 104 chunks based on their natural order (news source and time released). The N -best list used in this experiment is a list of 1000-best translations of 191 testing sentences (20 documents) extracted from the Chinese

MT04 test set by the Hiero system (Chiang 2005). For each one of the 191 testing sentence, we use each individual corpus chunk to re-rank its N -best list using the Q_{nc} feature. The difference of the re-ranked-best hypothesis and the model-best hypothesis in BLEU score is the *gain* of the corpus chunk denoted as $\mathcal{G}(t;d)$. $\mathcal{G}(t;d)$ can be negative if the re-ranked-best hypothesis has lower BLEU score than the model-best baseline translation.

Table 2.60 shows the predicted BLEU gain and the actual BLEU scores of using single features to select the most relevant corpus chunk for re-ranking. In general, higher order k -gram coverage rate predicts relevance better than lower-order coverage rate. However, from 5-gram coverage rate to 6-gram coverage rate we see a slight drop in predicted gain. Our explanation is that given the current SMT quality, 6-gram coverage rate is very low and a few “lucky” 6-gram hits are not representative of the corpus chunk. Results also show that k -gram coverage rate outperforms perplexity in predicting corpus relevance.

Relevance Feature	Gain	BLEU
$\sum_{r=1}^N Q_c(e^{(r)}; \mathcal{E})$	0.97	30.17
Coverage ₁ ($\mathcal{E}; \mathbf{f}$)	0.20	29.53
Coverage ₂ ($\mathcal{E}; \mathbf{f}$)	1.40	29.89
Coverage ₃ ($\mathcal{E}; \mathbf{f}$)	1.52	30.14
Coverage ₄ ($\mathcal{E}; \mathbf{f}$)	1.33	30.20
Coverage ₅ ($\mathcal{E}; \mathbf{f}$)	2.07	30.34
Coverage ₆ ($\mathcal{E}; \mathbf{f}$)	1.97	30.42
$\frac{1}{N} \sum_{r=1}^N Q_{l(e)}(e^{(r)}; \mathcal{E})$	1.06	30.37
$\frac{1}{N} \sum_{r=1}^N Q_{nc}(e^{(r)}; \mathcal{E})$	0.00	29.89
$PP(\mathbf{f} \mathcal{E})$	0.44	30.14

Table 2.60: Predicted BLEU gain and actual BLEU scores of using single LM feature to re-rank the N -best list.

Relevance features can be combined in a linear function to predict the relevance of a corpus chunk to a sentence. Table 2.61 shows the gain in BLEU score after re-ranking the N -best list using the most relevant corpus chunk selected based on manually set feature weights.

IBM Speech Recognition Results

We carried out experiments re-ranking the N -best lists output by a speech recognizer. The system description is similar to that given by Saon *et al.* (2005). The lattices are generated using a two-pass speech recognition system that performs a speaker-independent recognition pass; then a series of speaker adaptation steps, including vocal tract length normalization, feature-space MLLR and MLLR; and concludes with a speaker-adapted recognition pass. The speaker-independent and speaker-adapted acoustic models are trained on a 450-hour corpus comprising the 1996 and 1997 English Broadcast News Speech collections and the English broadcast audio from TDT-4. The language model used by the decoder was a 4-gram modified Kneser-Ney model trained on 192M words and interpolated with five other 4-gram models trained on topic specific subsets of the original 192M words data. The final merged model was pruned to a total of

3.2M n -grams. The final speaker-adapted recognition pass performed in-memory lattice re-scoring using a larger, 30M n -gram language model that differed from the one used to produce the decoding graph only in the degree of pruning.

Feature Combinations	Weights of k -gram Coverage Rate						BLEU Gain
	1	2	3	4	5	6	
1, 2-gram coverage rate	1.0	2.0					0.73
1, 2-gram coverage rate	-1.0	2.0					0.77
1, 2-gram coverage rate	-2.0	1.0					1.12
1, 2, 3-gram coverage rate	-2.0	1.0	2.0				0.74
1, 2, 3, 4-gram coverage rate	-2.0	1.0	2.0	3.0			1.27
1, 2, 3, 4, 5-gram coverage rate	-2.0	1.0	2.0	3.0	4.0		1.30
2, 3, 4, 5-gram coverage rate		2.0	3.0	4.0	5.0		1.11
2, 3, 4, 5-gram coverage rate		4.0	9.0	16.0	25.0		1.54
2, 3, 4, 5, 6-gram coverage rate		2.0	3.0	4.0	5.0	6.0	1.07
2, 3, 4, 5, 6-gram coverage rate		4.0	9.0	16.0	25.0	36.0	1.14
1, 2, 3, 4, 5, 6-gram coverage rate	-2.0	2.0	3.0	4.0	5.0	6.0	1.09
1, 2, 3, 4, 5, 6-gram coverage rate	1.0	4.0	9.0	16.0	25.0	36.0	1.14

Table 2.61: Feature weights set following the heuristic that higher order k -gram coverage rates have more discriminative power in choosing relevant corpus chunks.

	RT-03	Dev-04	RT-04
baseline	9.5	17.7	15.0
5-gram DLM	9.0	17.4	14.5

Table 2.62: ASR N -best re-ranking

Table 2.62 shows the results of using the distributed language model, in this case a 5-gram trained on 4 billion words, for re-ranking the 500-best lists from two DARPA's EARS program English Broadcast News evaluation sets (RT-03 and RT-04) and one development set (Dev-04). As can be seen, using the larger distributed LM results in considerable improvements in Word Error Rate across all three sets. It should be noted that out of the 500 hypotheses per utterance in average only 9.5 were unique word strings and the distributed model achieved the shown reduction by effectively only re-raking these remaining few hypotheses.

IBM Machine Translation Results

The phrase decoder we use is inspired by Tillmann and Ney (2003). It is a multi-stack, multi-beam search decoder with as many stacks as the length of the source sentence and a beam associated with each stack. It is described in more details by Al-Onaizan (2005). The training data for the Arabic translation system consisted of 4.6M sentence pairs, totaling 124M Arabic and 143M English words. The MT-03 and MT-05 test sets consist of 663 and 1056 segments, respectively. We also used a different decoder based on the Maximum Entropy (MaxEnt) approach. The MaxEnt decoder is a novel method of using simple translation blocks, which allows the translation problem to be cast into a classification problem where instead of encoding the context in long phrasal blocks, it is cast as feature function on simple blocks (Ittycheriah and Roukos 2007).

For the Chinese translation system CE Training data there were about 8.4M sentence pairs, totaling 212.5M Chinese and 241.2M English words. We only used the phrase-based decoder for the Chinese to English translation experiments. It should be noted that the training data was sampled for each test set; the actual training data being used is in average 400K sentences in size. The test sets MT-03, MT-04 and MT-05 are 919, 1788 and 1082 sentences long, respectively.

The results for the Arabic-English and Chinese-English experiments are given in Table 2.63. The 3-gram LM is a linearly interpolated model trained with a vocabulary size of 128K on about 2.8 billion words with bigram and trigram with less than 3 n -grams pruned. The 5-gram distributed model was trained on 2.3 billion words with a vocabulary size of just over a million. As can be seen, using the distributed model that allows for larger corpus and higher order n -gram with no pruning (not to mention larger vocabulary size) results in significant improvements across all systems and test set. For example, in the case of the Arabic-English system, the phrase-based decoder achieves a 3.3 BLEU point improvement in using a 5-gram language model over a 3-gram for both MT-03 and MT-05 test sets.

Model	Language	MT-03	MT-04	MT-05
Phrase-based Decoder 3gm	Arabic	48.84	-	46.92
Phrase-based Decoder 5gm DLM		51.10	-	50.25
MaxEnt Decoder 3gm		48.95	-	48.02
MaxEnt Decoder 5gm DLM		51.35	-	51.30
3-gram	Chinese	26.93	28.72	26.10
5-gram DLM		29.10	30.09	29.18

Table 2.63: BLEU Results

When compared with the ASR experiments, it is clear that we achieve better improvements with the distributed LM for the SMT task. One explanation is that for the SMT experiments, the distributed language model was directly integrated into the decoder, whereas in the ASR experiments the distributed model was used only to re-rank very shallow ($N \approx 9.5$) N -best lists. Another explanation is that the tokenization and vocabulary selection for the distributed language model were optimized with only the SMT tasks in mind.

2.5.1.7 Related work

On using LM to re-rank the N -best list, Kirchhoff and Yang (2005) used a factored 3-gram model and a 4-gram LM (modified KN smoothing) together with seven system scores to re-rank an SMT N -best. They improved the translation quality of their best baseline (Spanish-English) from BLEU 30.5 to BLEU 31.0.

On large language model work, Brants *et al.* (2007) use the MapReduce programming model to train on terabytes of data and to generate terabytes of language models. The trained n -gram information is distributed in a distributed cluster. New decoder architecture is implemented to access the n -gram information efficiently by first queuing certain number of LM requests (e.g., 1,000 or 10,000) and sends them in a batch to servers.

On language model adaptation work, Iyer and Ostendorf (1999) selected and weighed data to train language modeling for ASR. The data was selected based on its relevance for a topic or the similarity to the in-domain data. Each additional document is classified to be in-domain or out-of-domain according to cosine distance with TF-IDF term weights, POS-tag LM and a 3-gram word LM. n -gram counts from the in-domain and the additionally selected out-of-domain data are then combined with an weighting factor to estimate a LM with standard smoothing. Hildebrand *et al.* (2005) use information retrieval to select relevant data to train adapted translation and language models for an SMT system.

2.5.1.8 Conclusion

In this section we presented novel distributed architectures from IBM and CMU for storing large scale and scalable corpora and language models. The distributed architecture is capable of handling arbitrarily large corpora, while placing no restrictions on either the n -gram order or vocabulary size. Computing n -gram counts of arbitrary order was made possible by storing the original corpus in the suffix array format.

Three key features make these systems different from the traditional n -gram language models. First, the underlying suffix array language model allows them to use arbitrarily long history; second, the distributed computing architecture makes it possible to use arbitrarily large training data; and finally, the on-the-fly language model estimation facilitates the client to dynamically adapt the language model for each testing sentence. Furthermore, as more data becomes available it is quite straightforward to add them to the model as only the new data needs to be processed. In traditional n -gram models the entire counting and smoothing steps for all the training data needs to be repeated anytime one wished to add more data to the model.

Several different uses of the distributed architecture were discussed. IBM introduced a simple, but powerful heuristic smoothing algorithm to compute language model probabilities on-the-fly from the counts aggregated from the suffix array corpora. This heuristic algorithm was used in MT decoding and ASR N -best re-ranking experiments. CMU experimented with different sentence-level nativeness scores. They also exploited the flexibility of the distributed architecture, in terms of dynamically changing the weights on different corpora, to implement an online adaptive language model.

IBM applied the distributed language model in both MT decoding and N -best list re-ranking of a state-of-the-art ASR system. In both scenarios, consistent and considerable improvements in translation quality (for MT experiments as measured by the BLEU score) and transcription accuracy (WER in ASR experiments) were achieved. Similarly, CMU experiments have shown that the distributed suffix array language model can be used reliably and effectively for re-ranking the N -best list of the statistical machine translation system and significantly improves the translation quality.

2.6 Search and Complexity

2.6.1. Improvements for Beam Search in Statistical Machine Translation

Authors: Oliver Bender, Richard Zens and Hermann Ney

2.6.1.1 Introduction

In statistical machine translation (SMT), search (or decoding or generation) is the task of finding the target language sentence e_1^I that maximizes the posterior probability given the source sentence f_1^J . Here, we are considering a phrase-based approach with a segmentation s_1^K into K phrases (Och *et al.* 1999; Zens and Ney 2004; Koehn *et al.* 2003). Assuming a log-linear model with component models $h_m(\cdot)$ and scaling factors λ_m , $m = 1, \dots, M$, we obtain the decision rule:

$$\hat{e}_1^I = \operatorname{argmax}_{I, e_1^I} \{ \max_{K, s_1^K} \sum_{m=1}^M \lambda_m h_m(e_1^I, s_1^K; f_1^J) \}. \quad (2.48)$$

We have to carry out a maximization over all possible target sentences e_1^I and over all possible segmentations s_1^K . As enumerating all target sentences is infeasible, we are facing a hard optimization problem. In particular, we have to decide on:

- the number of phrases K ,
- the phrase segmentation s_1^K of the source sentence and
- the phrase translations \tilde{e} for each source phrase \tilde{f} .

However, we can exploit the structural properties of the models. We can interpret the search as a sequence of decisions (\tilde{e}_k, b_k, j_k) for $k = 1, \dots, K$. At each step, we decide on a source phrase \tilde{f}_k identified by its start and end positions $s_k = (b_k, j_k)$ and the corresponding translation \tilde{e}_k . To ensure that there are no gaps and no overlap, we keep track of the set of source positions that are already translated ('covered'). We call this the coverage set $C \subseteq \{1, \dots, J\}$ and refer to the number of covered source positions of a hypothesis as its cardinality c . Furthermore, the search space can be regarded as a graph where the arcs are labeled with the decisions (\tilde{e}, j, j') and the states are labeled with the coverage sets C . The initial state is labeled with the empty coverage set, i.e., no source word is yet translated and the goal state is labeled with the full coverage $C = \{1, \dots, J\}$. Each path through this graph represents a possible translation of the source sentence, simply by concatenation of the target phrases \tilde{e} along the path. Hence, the search problem can also be looked at as finding the optimal path through this graph.

Except for the language model (LM) and a distortion penalty model (DM), all the models of our decoder are defined on the level of single phrases, i.e., they have no dependencies across phrase boundaries and can, therefore, be computed for each phrase pair without context information. We group these models together and call them phrase models. With respect to the sequence of decisions (\tilde{e}_k, b_k, j_k) , the phrase models do not depend on the decisions taken so far. According to Zens and Ney (2008), we use

$q_{\text{TM}}(\tilde{e}_k, b_k, j_k)$ to denote the weighted sum of all phrase model scores for this decision (also called *state*). The LM on the other hand depends on the last (n-1) words of the target sentence and the DM depends on the end position of the previous phrase. Nevertheless, we want to compute the scores of individual states and therefore introduce state copies and distinguish them according to the history. We use $\tilde{e}' \otimes \tilde{e}$ to denote the LM history after expanding the given history \tilde{e}' with the phrase \tilde{e} . The corresponding expansion score weighted with the LM scaling factor is denoted as $q_{\text{LM}}(\tilde{e}|\tilde{e}')$. Accordingly, we use $q_{\text{DM}}(j, j')$ to denote the weighted score of a jump from source position j to source position j' . Obviously, (\tilde{e}_k, j, j') can occur multiple times in the search space. To avoid repeated computations, we determine the set of possible target phrases for all source phrases before the actual search and store the target phrases along with their scores in a table $E(j, j')$.

Summing up, the states in the search space can be identified by a triple (C, \tilde{e}_k, j) , where C denotes the coverage set, \tilde{e} denotes the LM history and j denotes the end position of the last source phrase. During search, a translation hypothesis is expanded by computing the successor states for the current state (C, \tilde{e}_k, j) . The expansion with a phrase pair (\tilde{e}', j'', j') yields the successor state $(C \cup \{j'', \dots, j'\}, \tilde{e} \otimes \tilde{e}', j')$ and the corresponding score is calculated as:

$$q_{\text{TM}}(\tilde{e}', j'', j') + q_{\text{LM}}(\tilde{e}'|\tilde{e}) + q_{\text{DM}}(j, j').$$

We have to ensure that there is no overlap, i.e. $C \cap \{j'', \dots, j'\} = \emptyset$.

2.6.1.2 Beam Search

As the size of the search graph is exponential in the source sentence length, we have to use approximations to find a solution efficiently. In fact, it has been shown by Knight, (1999) that the search problem is NP-hard. A recent exposition to the search problem was proposed by Jelinek (1998). Here, we resort to dynamic programming (Bellman 1957) and beam search (Jelinek, 1998) to address this problem. Using dynamic programming (DP), i.e., first computing small sub-problems and then assembling the solution for the whole problem from these sub-problems, we reduce the number of paths that we have to explore in the search graph. The idea of beam search is to keep the promising candidates and to discard hypotheses that are unlikely to yield the optimal solution. Thereby, beam search may generate suboptimal solutions. In the following, we make an explicit distinction between reordering and lexical hypotheses:

- **Coverage hypothesis** C . We use the term coverage hypothesis to refer to the set of all lexical hypotheses with the same coverage C .
- **Lexical hypothesis** (C, \tilde{e}, j) . A lexical hypothesis is identified by a coverage C , a language model history \tilde{e} and a source sentence position j .

The number of coverage or more intuitive reordering hypotheses indicates how many alternative reorderings per cardinality are investigated during the search. The number of lexical hypotheses per reordering hypothesis indicates the lexical alternatives that are taken into account.

2.6.1.2.1 Monotone Search

$$Q(0, \$) = 0$$

$$Q(j, \tilde{e}) = \max_{\substack{j': j-L_s \leq j' < j \\ \tilde{e}'', \tilde{e}': \tilde{e}' \oplus \tilde{e}'' = \tilde{e}}} \{Q(j', \tilde{e}') + q_{\text{TM}}(\tilde{e}'', j' + 1, j) + q_{\text{LM}}(\tilde{e}'' | \tilde{e}')\}$$

$$\hat{Q} = \max_{\tilde{e}} \{Q(J, \tilde{e}) + q_{\text{LM}}(\$; \tilde{e})\}$$

Figure 2.41: Dynamic programming recursion for monotone search.

If we prohibit phrase rearrangements during translation, both source and target sentence are processed in monotone order. There is no reordering of the phrases and the distortion penalty model becomes unnecessary. As a consequence, the monotone search problem can be solved efficiently using DP. For the decision rule given in Section 2.6.1.1, we define the quantity $Q(j, \tilde{e})$ as the maximum score of a phrase sequence that ends with the language model history \tilde{e} and covers positions 1 to j of the source sentence and obtain the DP recursion as shown in Figure 2.41. The $\$$ symbol denotes the sentence boundary marker and L_s denotes the maximum phrase length in the source language. During the search, we store back-pointers to the previous best decision $B(\cdot, \cdot)$ and to the maximizing arguments $A(\cdot, \cdot)$. After the search, the back pointers are used to trace back the best decisions and generate the translation. The resulting complexity is linear in the length of the source sentence, and thus, allows for a very efficient implementation.

The drawback is that reordering is only possible within the phrases (solely the sequence of phrases is enforced to be monotone). However, the monotone search algorithm is suitable for language pairs which have a similar word order, e.g., Arabic-English, as can be seen in Section 2.6.1.4.

2.6.1.2.2 Non-Monotone Search

If we tackle the translation problem for language pairs with a different word order, such as Chinese-English, the monotone search is inappropriate. Instead, we need to explicitly permit reordering. Tillmann and Ney (2003) described a non-monotone search algorithm for single word based translation. The idea is that the search proceeds synchronously with the number of already translated source positions. Therefore, they call their algorithm source cardinality synchronous search. Here, we use a phrase-based version of their method. We generate the translation phrase by phrase, i.e., the search is monotone in the target language. To permit reordering, we are allowed to jump back and forth within the source sentence. As one constraint of the phrase-based approach is that each source position is translated by exactly one target phrase, we have to keep track of the source positions already translated. Recapitulate that this corresponds to the coverage $C \subseteq \{1, \dots, J\}$.

In contrast to the monotone search, the auxiliary quantity for the DP recursion now also depends on the coverage set. $Q(C, \tilde{e}, j)$ denotes the maximum score of a path leading from the initial state to the state (C, \tilde{e}, j) . The corresponding DP recursion for the non-monotone search is given in Figure 2.42.

$$Q(\theta, \$, 0) = 0$$

$$Q(C, \tilde{e}, j) = \max_{\substack{j'', j': j' \leq j < j' + L_s \\ \wedge \{j', \dots, j\} \subseteq C \\ \tilde{e}', \tilde{e}'': \tilde{e}' \oplus \tilde{e}'' = \tilde{e}}} \{Q(C \setminus \{j', \dots, j\}, \tilde{e}', j'') + q_{TM}(\tilde{e}'', j', j) + q_{LM}(\tilde{e}'' | \tilde{e}') + q_{DM}(j'', j')\}$$

$$\hat{Q} = \max_{\tilde{e}, j} \{Q(\{1, \dots, J\}, \tilde{e}, j) + q_{LM}(\$ | \tilde{e}) + q_{DM}(j, J + 1)\}$$

Figure 2.42: Dynamic programming recursion for non-monotone search.

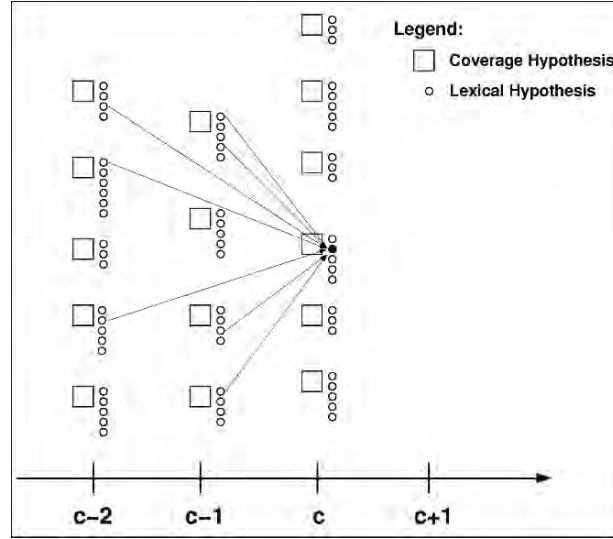


Figure 2.43: Illustration of the (non-monotone) search process. For each cardinality, we have a list of coverage hypotheses (boxes). For each coverage hypothesis, we have a list of lexical hypotheses (circles).

As for the monotone search, we store back pointers $B(\cdot)$ to the previous best decision as well as the maximizing arguments $A(\cdot)$. For each cardinality c , we have to iterate over all possible source phrase lengths l . Then, we have to account for all the possible predecessor coverages C' with cardinality $c - 1$. Next, we have to select a source phrase $\tilde{f} = f_j, \dots, f_{j+l}$ by choosing the start position j . Eventually, we consider all existing predecessor states \tilde{e}', j' and all translation options $\tilde{e}'' \in E(j, j + l)$ to compute the score of the expansion. In Figure 2.43, we illustrate the (non-monotone) search. For each cardinality, we have a list of coverage hypotheses, here represented as boxes. For each coverage hypothesis, we have a list of lexical hypotheses, here represented as circles. We generate a specific lexical hypothesis (the black circle) with cardinality c by expanding shorter hypotheses. The hypotheses with cardinality $c - 1$ are expanded with one-word phrases, the hypotheses with cardinality $c - 2$ are expanded with two-word phrases, etc. Consequently, all generated hypotheses have the same cardinality which allows for very efficient recombination and pruning. In contrast, e.g., Koehn *et al.* (2007) and Tillmann (2006) expand hypotheses with cardinality c into higher cardinalities.

In principle, we have to loop over all possible coverage sets which yields an exponential complexity of the non-monotone search algorithm, taking into account that

$\sum_{c=0}^J \binom{J}{c} = 2^J$. To make the translation process manageable, we resort to a beam search strategy and apply pruning at several levels.

2.6.1.2.3 Pruning

Our decoder implements two variants of pruning: threshold pruning (or beam pruning) and histogram pruning (Steinbiss *et al.* 1994). Threshold pruning secures that only hypotheses are kept whose scores are close to the best one. A drawback is that threshold pruning effects the beam size only indirect and that there is no upper limit on the number of hypotheses in the beam. Histogram pruning on the other hand means that only the best N hypotheses are kept and is thus, a very simple way of limiting the beam size. In this work, we mainly use histogram pruning. Threshold pruning is only partially applied, e.g., to tune the system in terms of translation speed.

The novel contribution of this work is that we make an explicit distinction between reordering and lexical pruning:

Reordering pruning:

The number of reordering hypotheses per cardinality c is limited. If we prune a reordering hypothesis, we remove all associated lexical hypotheses.

Lexical pruning:

The number of lexical hypotheses per reordering hypothesis is limited.

To be specific, we use the following pruning strategies.

Observation Pruning

Here, we limit the number of phrase translation candidates per source phrase. This is actually done before search. We apply observation histogram pruning with parameter N_o . Thus, if there are more than N_o target phrases for a particular source phrase, we keep only the top N_o candidates. In practice, $N_o = 50$ already achieves good results while keeping the phrase tables manageable, in particular for large-scale tasks such as GALE.

Lexical Pruning Per Coverage

Here, we consider all lexical hypotheses that share the same coverage C , differing for instance in their language model history \tilde{e} or the end position of the last phrase j . During pruning, we compare hypotheses which cover different parts of the source sentence. Hence, it is important to use a rest score estimate for completing these hypotheses. Without such a rest score estimate, the search would first focus on the easy-to-translate part of the source sentence. In this work, we deploy a rest score estimate $R(C, j)$ including translation, language and distortion model computed on sequences of source positions. A detailed description of the rest score estimate is given by Zens and Ney (2008). The rest cost estimate is of special importance for smaller beam sizes. Let τ_L denote the pruning threshold and let $Q(C)$ denote the maximum score of any hypothesis with coverage C :

$$C(Q) = \max_{\tilde{e}, j} \{Q(C, \tilde{e}, j) + R(C, j)\}.$$

Then, we keep a hypothesis with score if: $Q(C, \tilde{e}, j)$

$$Q(C, \tilde{e}, j) + R(C, j) + \tau_L \geq Q(C)$$

Generally, we apply histogram pruning with parameter N_L . Thus, if there are more than N_L hypotheses for a particular coverage C , then we keep only the top N_L candidates. In practice, $N_L = 500, \dots, 20\,000$ are typical values.

Coverage Pruning per Cardinality

Here, we consider all coverage hypotheses with a given cardinality c . $Q(C)$ is defined as the maximum score of any hypothesis with coverage C and is used here as score of the coverage hypothesis C . Let τ_c denote the pruning threshold, then we keep a coverage hypothesis with score $Q(C)$ if:

$$Q(C) + \tau_c \geq \max_{C': |C'|=c, \tilde{e}, j} \{Q(C', \tilde{e}, j) + R(C', j)\}.$$

Generally, we apply histogram pruning with parameter N_C . Thus, if there are more than N_C coverage hypotheses for a particular cardinality c , we keep only the top N_C candidates. Keep in mind that if we prune a coverage hypothesis C , we remove all lexical hypotheses with coverage C . In practice, N_C is in the range of 5 to 50.

2.6.1.3 Search Algorithm

Due to histogram pruning, the computational complexity of the non-monotone search is now also linear in the sentence length (although encountering a rather large constant factor). On the other hand, the search is now no longer guaranteed to find the (global) optimal translation candidate. However, the experimental results show that the pruning strategies almost cause no loss in translation quality. Section 2.6.1.4 also stresses the importance of the coverage pruning to adjust the balance between hypotheses representing different reorderings (coverage hypotheses) and hypotheses with different lexical representations.

Putting everything together, the resulting pseudo code for the non-monotone search algorithm including pruning is:

```

INPUT: source sentence  $f_1^J$  translation options  $E(j, j')$  for  $1 \leq j \leq j' \leq J$ ,
       models  $q_{TM}(\cdot)$ ,  $q_{LM}(\cdot)$  and  $q_{DM}(\cdot)$ 
0    $Q(\emptyset, \$, 0) = 0$ ; all other  $Q(\cdot, \cdot, \cdot)$  entries are initialized to  $-\infty$ 
1   FOR cardinality  $c = 1$  TO  $J$  DO
2     IF  $c > L_S$  THEN purgeCardinality  $c - L_S - 1$ 
3     FOR source phrase length  $l = 1$  TO  $\min\{L_S, c\}$  DO
4       FOR ALL coverages  $C' \subset \{1, \dots, J\} : |C'| = c - l$  DO
5         FOR ALL start positions  $j \in \{1, \dots, J\} : C' \cap \{j, \dots, j + l\} = \emptyset$  DO
6           coverage  $C = C' \cup \{j, \dots, j + l\}$ 
7           FOR ALL states  $\tilde{e}', j' \in Q(C', \cdot, \cdot)$  DO

```



```

8      partial score  $q = Q(C', \tilde{e}', j') + q_{DM}(j', j)$ 
9      IF  $q + R(C, j + l) + q_{TM}(j, j + l)$  isTooBadForCoverage  $C$ 
10     THEN CONTINUE
11     FOR ALL phrase translations  $\tilde{e}'' \in E(j, j + l)$  DO
12       partial score  $q' = q + R(C, j + l) + q_{TM}(\tilde{e}'', j, j + l)$ 
13       IF  $q'$  isTooBadForCoverage  $C$  THEN BREAK
14       score =  $q + q_{TM}(\tilde{e}'', j, j + l) + q_{LM}(\tilde{e}'' | \tilde{e}')$ 
15       IF score +  $R(C, j + l)$  isTooBadForCoverage  $C$  THEN CONTINUE
16       language model state  $\tilde{e} = \tilde{e}' \otimes \tilde{e}''$ 
17       IF score >  $Q(C, \tilde{e}, j + l)$ 
18       THEN  $Q(C, \tilde{e}, j + l) = \text{score}$ 
19         B  $(C, \tilde{e}, j + l) = (C', \tilde{e}', j')$ 
20         A  $(C, \tilde{e}, j + l) = \tilde{e}$ 
21     pruneCardinality  $c$ 

```

The basic concept of the algorithm is the same as for the monotone search. A key challenge is to perform as much computations outside the inner loops as possible and to apply pruning wherever applicable. The function 'pruneCardinality c ' applies coverage and cardinality pruning after all hypotheses with the current cardinality c have been generated (line 21). In the function 'purgeCardinality c ', we free the memory (except trace back information) of all hypotheses with cardinality c . For example, the coverage sets and the LM histories are not needed anymore, and thus, the memory can be reused. Furthermore, we can stop the expansion whenever it is clear that the resulting hypotheses would be pruned anyway. This is done via the 'x isTooBadForCoverage C ' function. As the translation options $E(\cdot, \cdot)$ are sorted once before the search, we can process the hypotheses according to their score and check:

- if the partial score plus rest cost estimate plus an optimistic estimation for the translation model score is too bad, we can skip all of the possible phrase translations (line 9),
- if the partial score without LM is already too bad, we can omit the LM score computation (line 13),
- if the score of the expansion is too bad, we can ignore to check for recombination (line 15).

The presented algorithm is comparable with the cube pruning approach of Chiang (2005) and Huang and Chiang (2007). We observed that there is virtually no difference between cube pruning and our implementation, but the benefits of cube pruning, i.e., fewer LM expansions, can be achieved by simpler means as shown in this work. Early discarding of likely weak translation candidates and the effect of including the distortion rest score is equivalent to the method presented by Moore and Quirk (2007).

2.6.1.4 Experimental Results

We carried out experiments on the Chinese-English and Arabic-English GALE MT from text tasks (2007 development sets), i.e. translating texts out of the newswire (NW) and web domain (WT). The corpus statistics are shown in Table 2.64. Additional monolingual data, e.g. GigaWord v2, TDT, etc., were used for LM training. We applied modified Kneser-Ney discounting as provided by the SRILM toolkit (Stolcke 2002). For both tasks, translation results in terms of BLEU (Papineni *et al.* 2002) and TER (Snover *et al.* 2006) scores are reported in Table 2.65. These scores were calculated case insensitive and w.r.t single references. The reordering limit in all experiments was 10 words.

		Chinese	English	Arabic	English
Train	Sentence pairs	8.7M		7.4M	
	Running words	230 M	230 M	186 M	188 M
	Vocabulary size	242K	242K	565K	506K
Test	NW	Sentences	554		686
		Words	19K	21K	27K
	WT	Sentences	657		907
		Words	18K	20K	23K

Table 2.64: Corpus statistics of the Chinese-English and Arabic-English GALE MT from text task (2007 development sets).

	BLEU	TER
CE NW	18.66	68.29
CE WT	15.62	73.01
AE NW	29.60	54.75
AE WT	18.92	73.88

Table 2.65: Translation performance for GALE MT from text tasks (2007 development sets)

In Figure 2.44 and Figure 2.45, we separate the effect of the number of lexical and reordering hypotheses on the translation quality. The plots demonstrate that already a small number of lexical alternatives is sufficient to achieve good translation quality. For each curve, we limited the number of reordering hypotheses and varied the maximum number of lexical hypotheses per reordering hypothesis. Thus, along the x-axis we increased the search space by allowing for more lexical choice, whereas from curve to curve we allowed for more reordering. The overall search space is limited by the product of the two numbers, i.e. we varied the beam size from 1 to 64K. Both figures point out that there is no benefit from increasing the number of lexical choices beyond 16 candidates per reordering hypotheses. If we look at the maximum number of reordering choices, we see that reordering seems to be only a minor problem for the Arabic-English language pair. In combination with the implicit within phrase word reorderings, just 4 reordering alternatives times 16 lexical hypotheses for each of them are sufficient to exhaust the search space. By contrast, the maximum number of coverage hypotheses has a much bigger effect on the BLEU score for the Chinese-English translation direction. There is a considerable improvement by increasing the number of coverage hypotheses

up to 64. Furthermore, the improvement achieved by taking more reordering alternatives into account exceeds the improvement due to more lexical choices.

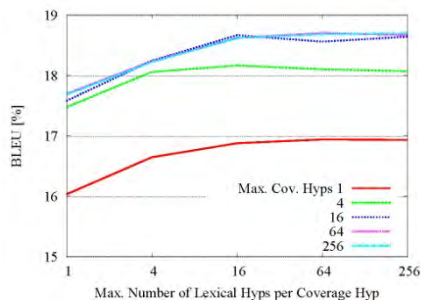


Figure 2.44: Effect of the number of lexical and reordering hypotheses for the Chinese-English newswire task.

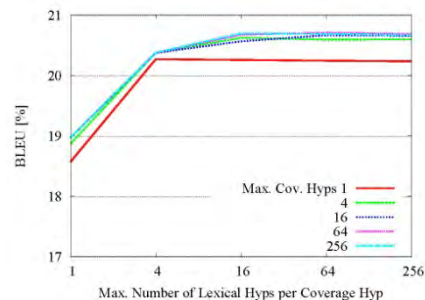


Figure 2.45: Effect of the number of lexical and reordering hypotheses for the Arabic-English web text task.

Finally, we performed a comparison with Moses (Koehn *et al.* 2007), a publicly available decoder. We used identical settings (TM, LM, λ_m , etc.) to translate the test data with varying beam sizes and measured the BLEU score and the speed of the two decoders. We observed that we can achieve the same BLEU score as Moses at a higher translation speed. Since we used a different subset of our MT development data drawn from the newswire domain, the numbers are not directly linked to the ones of Table 2.65 and Figure 2.44. Nevertheless, by looking at Figure 2.46, we found out that our decoder is faster by a factor of 10 to 20, e.g., if we look at the BLEU score level of 36.9%, our decoder (225.3 words per sec) is even 23 times faster than Moses (9.8 words per sec). These numbers indicate that our decoder is significantly more efficient than Moses, which makes our decoder appealing for online translations.

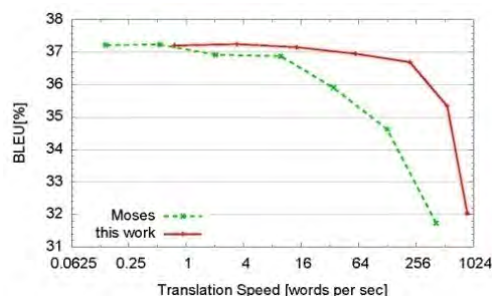


Figure 2.46: Comparison with Moses.

2.6.1.5 Conclusion

We presented an efficient search algorithm for phrase-based SMT. At first, we analyzed the search problem in detail and then developed the algorithm resorting to dynamic programming and beam search. The experimental results pointed out that it is important to keep alternative reordering hypotheses in the beam for the Chinese-English

language pair. On the other side, already a small number of lexical alternatives per reordering yields translations of good quality. A comparison with Moses emphasized the efficiency of the presented implementation, as our decoder is significantly faster at the same level of translation quality.

2.6.2. Handling Complexity in Decoding

Author: Christoph Tillmann

2.6.2.1 Introduction

This section presents several extensions of a commonly used phrase-based decoder for SMT (Koehn 2004a; Och and Ney 2004).²⁵ Such decoders are widely used within the GALE project and this section demonstrates ways of making these decoders more efficient in a principled way. It follows work by Tillmann and Ney (2003), in which the word reordering problem in SMT is linked to the traveling salesman problem (TSP). In that paper, a DP-based (dynamic programming-based) optimization algorithm to solve the TSP (Held and Karp 1962) serves as a starting point to handle the word reordering problem efficiently. The Held-Karp algorithm is a special case of a shortest path finding algorithm in a directed acyclic graph (DAG) (Dreyfus and Law 1977). Following this line of argumentation, this section spells out an approach that handles SMT decoding as a shortest path finding problem in dag's in some detail.

The translation model used in this section is a so called block-based model. A block is a pair of phrases which are translations of each other. During decoding, we view translation as a block segmentation process, where the input sentence is segmented from left to right and the target sentence is generated one phrase at a time. For some language pairs, close to state-of-the-art performance can be obtained by generating a largely monotone block sequence. A block sequence is scored as follows:

$$s_w(b_1^n) = \sum_{i=1}^n w^T \cdot f(b_i, b_{i-1}). \quad (2.49)$$

We try to find a block sequence b_1^n that minimizes $s_w(b_1^n)$ under the restriction that the concatenated source phrases of the blocks b_i yield a segmentation of the input sentence. We use the following block bigram scoring: a block pair $(b; b')$ is represented as a feature vector $f(b; b') \in \mathfrak{R}^N$. w is the feature weight vector which is trained on some development set. The feature vector components are the negative logarithm of some probabilities. Under this view, SMT becomes quite similar to sequential natural language annotation problems such as part-of-speech tagging, phrase chunking and shallow parsing.²⁶

²⁵ The work presented in this section is a compilation of work that has been published previously by Tillmann (2001, 2006, 2008) and Tillmann and Ney (2003).

²⁶ Since standard phrase-based decoders rely mostly on locally computed feature functions, the decoding algorithms presented in this section are applicable to them as well.

2.6.2.2 Beam-Search Decoder

In DP-based decoders for SMT, the search space is dynamically constructed: suitably defined lists of path hypotheses are maintained. This section focuses on some details how these path hypotheses are managed: using two stacks or multiple stacks. The standard phrase-based decoder by Koehn (2004a) uses J hypotheses stacks, one stack for each source position in the input sentence. The algorithm finds the Viterbi block translation for the model defined in Equation 2.49. Search states are 3-tuples of the following type:

$$[C, h ; d] . \quad (2.50)$$

$h = ([j, j'], [u, v])$ is the state history, where $([j, j'])$ is the interval where the most recent source phrase matched the input sentence and $[u, v]$ are the final two target words of the partial translation produced so far. d is the partial translation cost. Here, we use a coverage vector C to ensure that each source position is covered exactly once. That way, we make sure that a consistent alignment is produced. By including a coverage vector C into the search state definition we obtain an inherently exponential complexity: for an input sentence of length J there are 2^J coverage vectors (Koehn 2004a). By restricting the coverage vector appropriately, the decoding can be carried out efficiently as shown in Section 2.6.2.3. When a state of the type in Equation 2.50 is extended by some additional phrase match, a local transition cost Δ is computed. The partial cost d' of the new state is computed as $d' = d + \Delta$. If the original state covers k source positions and the latest source phrase match is l words long, the new hypothesis is added into the $(k + l)$ -th stack.

The second implementation uses two stacks to keep a single beam of active states. This corresponds to a beam-search decoder in speech recognition, where path hypotheses that correspond to word sequences are processed in a time-synchronous way. At a given time step the decoder keeps only hypotheses within some percentage of the best hypothesis (Lowerre and Reddy 1980; Ney *et al.* 1992). The state definition in Equation 2.51 includes an additional field l that keeps track of how much of the most recent source phrase match has been covered.

$$[C, l, h ; d] . \quad (2.51)$$

When a phrase is matched to the input sentence, the pointer l is set to position j where the most recent block match starts. As long as the pointer l is not yet equal to the match end position j' , it is increased: $l := l + 1$. Here, the single-beam decoder processes hypotheses *cardinality-synchronously*, i.e., states that cover k source positions generate new states that cover $k + 1$ positions. The local transition cost Δ' is spread evenly over the span of the latest match of length l : $\Delta' = \Delta/l$. The algorithm stores the states in only two stacks Γ and Γ' : Γ contains the most probable hypotheses that were kept in the last beam pruning step all of which cover k source positions. Γ' contains all the newly generated hypotheses before they are pruned in the next pruning step.

Before expanding a state set, states are pruned based on their coverage vector and the path cost: two different pruning strategies are used that have been introduced by Tillmann and Ney (2003) *coverage pruning* prunes states that share the same coverage vector C , *2) cardinality pruning* prunes states according to the number of covered positions k : all

states in the beam are compared with each other. The size of the beam depends locally on the number of hypotheses whose score is close to the top scoring one. In particular the cardinality pruning becomes an efficient pruning strategy in the case of the two-stack decoder: all active hypotheses are compared against each other. Adding a state to a stack includes adding the state if it is not yet present or updating its shortest path cost d if the state is already in that stack. This operation is also called *recombination*. It is a way to prune the stack safely without the risk of additional search errors. The recombination becomes very efficient if the coverage vector is restricted as shown in the next section. The multiple stack and two stack algorithm search slightly different search graphs where the single-beam algorithm with the more ‘regularly’ structured search graph results in a more efficient pruning strategy. The results of Tillmann (2006) show that using two stacks results in a runtime reduction of a factor of two when compared to a standard decoder for phrase-based SMT that uses a single stack for each source position.

2.6.2.3 Shortest Path Algorithm for Directed Acyclic Graphs (DAG)

This section compares the phrase decoding algorithms to a shortest path finding algorithm for Directed Acyclic Graphs (DAG). A DAG $G = (V, E)$ is a weighted graph for which a topological sort of its vertex set V exists: all the vertices can be enumerated in linear order. For such a weighted graph, we can compute the shortest path from a single source in $O(|V| + |E|)$ time, where $|V|$ is the number of vertexes and $|E|$ number of edges in the graph.

Here, vertices in the DAG correspond to the states defined in Equation 2.50 and edges correspond to transitions between states. The vertex set is finite since there is only a finite number of states. The corresponding graph contains no loops by definition: state transitions exist only from a state that covers fewer source positions to a successor state that covers additional (more) source positions. The fact that states are generated by increasing coverage cardinality can be understood as computing a topological sort of the vertices on the fly. The analysis in terms of a DAG shortest path algorithm can be used for a simple complexity analysis of the proposed algorithms. Local state transitions correspond to an edge traversal in the DAG search algorithm. These involve costly look-up operations, e.g., language, distortion and translation model probability lookup. Typically the computation time for update operations on lists Γ is negligible compared to these probability lookups’. That way, the search algorithm's complexity is simply computed as the number of edges in the search graph: $O(|V| + |E|) \approx O(|E|)$. Such an analysis has been presented by Tillmann (2001) for a series of reordering restrictions. In the case that the number of holes in the coverage vector for a left-to-right traversal of the input sentence is bounded by a constant k , e.g. $k = 3$, Tillmann (2001) gives a proof that the complexity for the resulting search algorithm is bounded by J^{k+1} where J is the length of the input sentence (the complexity due to the use of a language model or distortion model is ignored).²⁷ It is interesting to note that these complexity bounds are achieved

²⁷ Tillmann’s analysis (2001) is carried out for a word-based search where search states are of the form $[C, j]$ and j denotes the last covered source position j . The search state definition for a phrase-based decoder might result in a slightly higher complexity bound, but a dag complexity analysis remains feasible.

without the typical reordering window size restriction. Designing algorithms with provable upper complexity bounds might be a way to obtain more efficient decoding algorithm in the future.

The stack-based decoding algorithms can also be compared to an Earley-style parsing algorithm that processes lists of parse states in a single left-to-right run over the input sentence. That parsing algorithm can also be analyzed in terms of a DAG (Jurafsky and Martin 2000). Furthermore, the use of the pointer l in the state definition in Equation 2.51 is related to the use of the so-called dotted rules in the Earley parser states.

2.6.2.4 POS-based Re-order Rules

The section sketches an extension of the beam-search decoder introduced in Section 2.6.2.2, where POS-based reorder rules (Crego and Marino 2006) are tightly integrated into a single-beam left-to-right search. Reorder rules are typically applied to the source and/or target sentence as a preprocessing step (Xia and McCord 2004; Collins *et al.* 2005; Paulik *et al.* 2007). A few dozen reorder rules generate a so-called reordering graph. In comparison, the novel rule-driven decoder can handle about 30,000 rules efficiently. The algorithm described in this section uses an intermediate data structure called an *edge* that represents a source phrase together with its target phrase translation (A similar data structure is called *translation option* by Koehn (2004a)). For each input interval that is matched by some source phrase, we store a list of possible target phrase translations as edges in a *chart*. Additionally, we replace the reorder graph generation by a simpler edge generation process which involves only local computation. To formalize the approach, the search state definition in Equation 2.50 is modified as follows:

$$[s; [i, j], r, s_r, e \in \{\text{false}, \text{true}\}] \quad (2.52)$$

Here, the coverage vector C is replaced by a single number s : a monotone search is carried out and all the source positions up to position s (including s) are covered. $[i, j]$ is the coverage interval for the last source phrase translated (the same as in Equation 2.50). r is the rule identifier, s_r is the starting position for the match of the r^{th} rule in the input sentence and e is a flag that indicates whether the hypothesis h has covered the entire span of rule r yet.²⁸

States are extended by finding matching edges: simple or rule edges. We illustrate the generation of these edges in Figure 2.47 for the use of two overlapping rules on a source segment of four words a_0, \dots, a_3 . The two rules correspond to matching POS sequences, i.e., the input sentence has been POS tagged and a POS tag p_j has been assigned to each word a_j . Edges are shown as rectangles. In the top half of the picture, six simple edges which correspond to six phrase-to-phrase translations are enumerated. In the bottom half of Figure 2.47, we show the newly generated rule edges. A rule edge is generated from a simple edge via a reorder rule application and we add it into the chart. Here, rule 1

²⁸ For technical details, see the work of Tillmann (2008).

generates two new edges and rule 2 generates four new edges. For a consistent processing of the rule edges within the rule match interval, they are assigned one out of three positions: *BEGIN*, *INTER* (*mediate*) and *END*. We carry out the edge generation process in the following way: for each source interval $[k, l]$ all the matching phrase pairs are found and added into the chart as simple edges. In a second run over the input sentence for each source interval $[k, l]$, all matching POS sequences are computed and the corresponding source words $a_k \dots a_l$ are reordered according to the rule permutation. On the reordered word sequence phrase matches are computed and added into the chart. This edge generation is computationally 'light-weight' and typically takes only a few percent of the overall decoding time. Finally, the rule-driven decoder searches through the DAG defined by both simple and rule edges to find the best scoring block translation sequence.

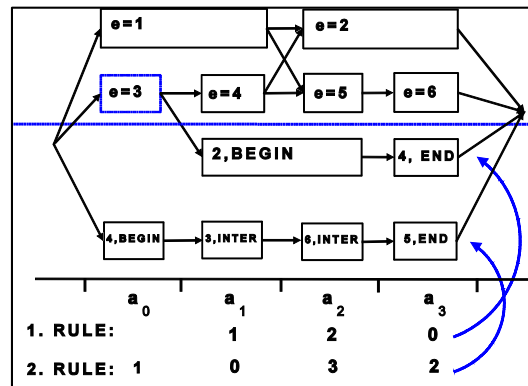


Figure 2.47: Addition of rule edges to a chart containing 6 simple edges. Rule application results in the generation of special rule edges. These edges are processed in an order consistent with the rule.

It is important to point out that the coverage vector component has been removed altogether from the search state definition in Equation 2.52. This way, the rule-driven search results in an essentially linear time decoding algorithm by 'indexing' the search space according to the rules and making extended use of the basic edge data structure. The rule-driven decoder uses the two stack implementation shown in Section 2.6.2.2. Here, we use an additional probabilistic feature, which is derived from the rule unigram count $N(r)$. Tillmann (2008), compares the rule-driven decoder to a standard phrase-based decoder. It is shown that decoding can be carried out efficiently and reliably with a large reordering window of 15 words: a small improvement in terms of BLEU is reported for a standard Arabic-English translation task. In addition, the rule-driven decoder handles a huge set of about 30,000 rules efficiently.

2.6.2.5 Efficient Block Segmentation Algorithm

A common approach to phrase-based SMT is to learn phrasal translation pairs from word-aligned training data. In this case, one uses a symmetrization step (Och and Ney 2004) and a word-alignment is computed for both translation directions source-to-target and target-to-source and the intersection of these two alignments is computed to obtain a high-precision word alignment. Here, we note that if this intersection covers all source

and target positions, then it constitutes a bijection between source and target sentence positions.²⁹ Additionally, Och and Ney (2004) define the notion of *consistency* for the set of phrasal translations that are learned from word-aligned training data; when projecting source intervals to target intervals and target intervals to source intervals no alignment link may lie outside the resulting phrase link. In Figure 2.48, we show examples of consistent and inconsistent phrase pairs. The word alignment for that sentence pair links $J = 4$ source and $I = 4$ target words. The alignment links that yield a bijection are shown as solid dots. When the alignment A is a bijection, by definition each target position i is aligned to exactly one source position j and vice versa and source and target sentence have the same length.

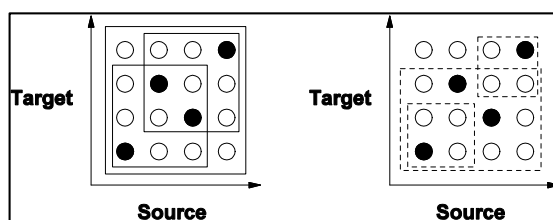


Figure 2.48: The left picture shows three consistent phrase pairs while the right picture shows three non-consistent phrase pairs.

We show that an efficient DP-based phrase alignment algorithm exists if the phrase pairs involved are consistent with an underlying alignment bijection. Without an underlying alignment \mathcal{A} , each pair of source interval S and target interval T defines a possible phrase link and this unrestricted phrase alignment problem has been shown to be NP-complete (De Nero and Klein 2008). The consistency restriction drastically reduces the number of phrase pair links. Source and target sentence segmentations are linked in a perfect way. Because of the bijection and the consistency restriction, target interval segmentation alone is sufficient to determine the source interval segmentation **and** the phrase alignment simultaneously. This close link is illustrated in Figure 2.49. The DP algorithm that makes use of the observation above uses the following auxiliary quantity:

²⁹ Tillmann (2003) reports an intersection coverage of about 65 % for Arabic-English data and a coverage of 40 % for Chinese-English data under real conditions.

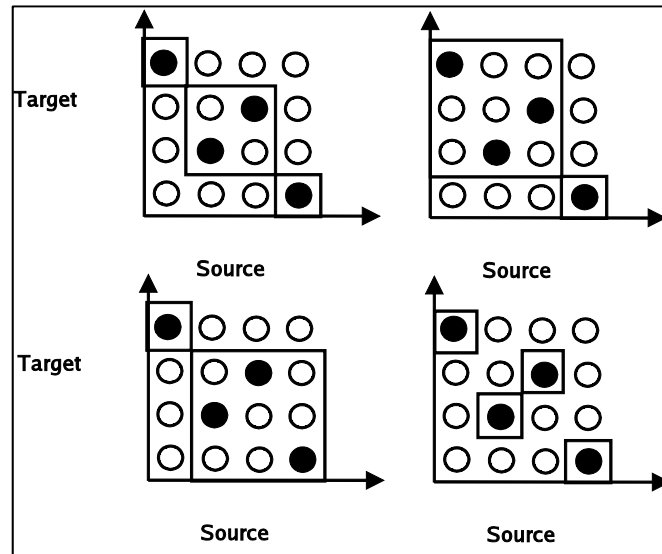


Figure 2.49: Four block segmentations of a sentence pair if the intersection is a bijection. The segmentation that covers the whole sentence pair with a single block is not shown.

$Q_b(i) :=$ score of the best partial segmentation that covers the target interval $[1, i]$ and ends in block b . The DP-based algorithm to compute $Q_b(i)$ is:

```

input: Parallel sentence pair and alignment  $\mathcal{A}$ .
initialization:  $Q_b(i) = -\infty$  for all  $i = 0, \dots, I$  and blocks  $b$ .
Sentence initial cost  $Q_{\hat{b}}(0) = 0.0$ .
for each  $i = 1, 2, \dots, I$  do
  for each  $i' = 0, 1, 2, \dots, i-1$  do
    Project interval  $T = [i' + 1, i]$  and check block link  $(T, S)$  for consistency.
    if  $(T, S)$  is CONSISTENT
      Set block  $b = (\text{words in } T, \text{words in } S)$ 
       $Q_b(i) = \max_{b'} \Delta(b, b') + Q_{b'}(i')$ 
recover optimal segmentation  $b_{i^n}$  :
  - find best end hypothesis:  $\max_b Q_b(I)$ 

```

A block segmentation is guaranteed to be found: the block that covers all source and target position is consistent by definition. Target intervals are processed from bottom to top. A target interval $T = [i', i]$ is projected using the word alignment \mathcal{A} , where a given target interval might not yield a consistent phrase pair, e.g. in Figure 2.49, out of all eight possible target segmentations, only five yield segmentations with consistent block links. For the initialization, we set $Q_b(i) = -\infty$ for all target positions $i = 0, \dots, I$ and blocks b and cost $Q_{\hat{b}}(0) = 0.0$, where \hat{b} is a special starting block positioned at the sentence beginning. The final score is obtained as $Q_{Final} = \max_b Q_b(I)$, where I is the length of the target sentence. $\Delta(b, b')$ denotes a transition cost for placing consecutive blocks which might use the same baseline features as in Section 2.6.2. Depending on the features that are being used, the auxiliary quantity in $Q_b(i)$ needs to be adjusted, e.g., by including source matching intervals to handle distortion features. The complexity of the algorithm

is $O(|B| \cdot I^2)$ assuming that the transition cost $\Delta(b, b')$ between adjacent blocks can be computed in constant time. Here, $|B|$ denotes the number of blocks that can be extracted from the aligned sentence pair. The presented DP-based algorithm is interesting theoretically because it solves a NP-complete problem efficiently (under the consistency restriction) for arbitrary long sentence pairs and unrestricted phrase-level reordering.

2.6.2.6 Discussion

Future work might bring the DP algorithms in Section 2.6.2.2 and Section 2.6.2.5 together: if the alignment intersection covers a source and target sentence only partially, the number of consistent phrase pairs is increased significantly. A stack-based search can be used to find a phrase alignment efficiently under the consistency restriction. Such an algorithm might be helpful in training a phrase-based system discriminatively. The single-pass, left-to-right beam search algorithms presented in Section 2.6.2.2 and Section 2.6.2.4 might be extended to decode with hierarchical model parameters. Under appropriate restrictions, this might result in a more efficient decoding algorithm when compared to a standard chart-based decoder.

2.6.3. Lattice Decoding for Hierarchical Models

Authors: Philip Resnik, Chris Dyer and Smaranda Muresan

2.6.3.1 Introduction

In any machine translation system, a number of choices are made at the outset. Typically these include aspects of source language representation such as segmentation, morphological analysis and orthographic normalization and are categorized as decisions about preprocessing. It is natural to ask which choices lead to better translations. Is it better to separate Arabic clitics into distinct tokens? How aggressive should the Chinese segmenter be? Unfortunately, the answer to questions like these is, fairly universally, “it depends”. For example, Habash and Sadat (2006) show that the optimal Arabic preprocessing approach is highly dependent on the amount of training data available. As a result, in the absence of useful generalizations, experimenters must resort to labor-intensive experimentation on development data, make a choice and hope that it is the right choice when the test data come along.

In this section, we describe a new approach that replaces static preprocessing decisions with dynamic representational decisions taken at runtime. The intuition is straightforward: the input representation to the decoder should capture all the possible decisions, rather than committing to a single resolution of preprocessing ambiguity and the decoder should be equipped to make relevant choices taking into account all the information available to it. In principle (and, as we will show, in practice), the “right choice” might in fact vary from word to word within a sentence being translated. The algorithms we present are inspired by recent developments in speech translation, where preserving ambiguity about the words that were spoken yields better performance than translating the 1-best transcription. We show how decoding with hierarchical phrase-based models (Chiang 2007) can be extended first to handle confusion networks, which

admit particularly efficient dynamic programming algorithms at the cost of some over generation (Dyer 2007a), and then to handle weighted lattices in the general case (Dyer *et al.* 2008). We briefly summarize relevant GALE experimentation, demonstrating significant improvements in translation quality and we close by discussing the promise of the approach for future work.

2.6.3.2 Preprocessing and Ambiguity

Because state-of-the-art statistical MT methods focus on learning source/target mappings from parallel text, the quality of translation depends heavily on language-specific processing. In any MT system, such preprocessing turns out to be surprisingly important, sometimes bearing as much responsibility for system performance (or lack thereof) as the scale of the training data or the cleverness of the models.

Preprocessing issues need to be addressed for any source and target languages and conceptually, these issues bear a strong family resemblance to problems associated with index term extraction in cross language information retrieval (Levow *et al.*, 2005). For IR systems, the goal is to characterize a document's content as a collection of discrete units that map sensibly to query terms. For MT systems, the goal is to represent the source and target sentences as token sequences for which correspondences of meaning are captured well by means of token-level alignments. Examples pointed out by Levow, Oard and Resnik are just as pertinent for MT as for IR: splitting clitics in Romance languages (*m'aidez* \Rightarrow *m' aidez* \Rightarrow *me aidez*), bound morpheme segmentation in Arabic (*wktAbAn* \Rightarrow *w ktAb An* [and two books]), Chinese word segmentation and German decomposing (*finanzminister* \Rightarrow *finanz minister* [minister of finance]). And, of course, for all languages, particularly in the newswire genre, the quality of translation can depend heavily on the treatment of numbers, dates and named entities.

A common approach, really the path of least resistance, is to preprocess the source sentence via 1-best analysis and if multiple analyzers are available, to run experiments on development data in order to select a single best analyzer whose 1-best output will be used. For example, Habash and Sadat (2006) explore a variety of approaches to Arabic morpheme splitting for MT. Each segmentation scheme is applied to the full training data and test data and the one performing best on the full test set is identified. However, this approach fails to take into account the fact that different analyzers may perform well on different subsets of the data.

One could therefore imagine treating different subsets of the data differently in terms of their preprocessing. This could include distinctions at the document level, e.g., training different systems, with different preprocessing approaches for newswire versus Web materials. Or, one could imagine going further, taking a supervised classification approach and directing different input sentences to MT system variants that use the preprocessing choices most likely to lead to good translations.

In this work, we have taken that intuition a step further, in keeping with the logical question it raises: why should the selection of the best preprocessing path even be made at the sentence level? It might well be that the quality of preprocessing choices varies across individual phrases within a single sentence. This motivates an approach in which sub-sentential preprocessing alternatives are preserved in the representation of the source

sentence, with statistical modeling/learning enabling the best possible combination of choices spanning the entire sentence.

2.6.3.3 Preserving Ambiguity Using Confusion Networks

Confusion networks are a compact representation of ambiguity that were introduced in speech recognition (Mangu *et al.* 2000) and have proven useful in phrase-based statistical machine translation (Bertoldi *et al.* 2007). In this section, we briefly characterize confusion networks in formal terms, then show how they can be handled as input to a hierarchical phrase-based decoder and finally summarize some experimental results that demonstrate their value.

2.6.3.3.1 Definition

A confusion network \mathcal{G} is a weighted directed acyclic graph with words labeling its edges, a single start node and a single end node. Unlike conventional word lattices, which can describe any finite set of strings, CNs are constrained such that every path in \mathcal{G} goes through every node in the same order. This allows \mathcal{G} to efficiently represent a very large number of sentences and permits the definition of very efficient dynamic programming algorithms for decoding.

Formally, \mathcal{G} can be described as a tuple of a word matrix \mathbf{F} and a posterior probability matrix \mathbf{p} , where $\mathbf{p}_{i,j} = P(\mathbf{F}_{i,j} | i, \mathbf{o})$. Traditionally, \mathbf{o} represents observations of the acoustic signal in a speech recognizer, but in our setting it arises from preprocessing ambiguity. We refer to the column of words at \mathbf{F}_i as the alternatives at position i .³⁰

1	2	3	4	5	6	7
sAfr 0.8	Al- 0.9	r}ys 0.9	l- 0.5	Amryky 0.9	EIY 0.3	bgdAd 1.0
sfyr 0.2	ε 0.1	ε 0.1	Al- 0.4	AmrykA 0.1	lA 0.3	
			ε 0.1		AlY 0.2	
					fy 0.1	
					ε 0.1	

Table 2.66: Example confusion network. A CN has a fixed length and each column has a distribution over words that may appear in that position. This example shows some ambiguities typical of an ASR system for the Arabic sentence *sAfr Alr}ys AlAmryky AlY bgdAd* (in English, *the American president traveled to Bagdad*).

Table 2.66 illustrates a confusion network that would be typical as output from an Arabic ASR system. Arabic words are given in the Buckwalter convention. In this artificial example, the 1-best transcription hypothesis (top row) contains two mistakes. The first mistake is at position 4, where the correct word, the definite article *Al*, has lower

³⁰ In order to describe sentences that are arbitrarily shorter than the length of the paths through the network, \mathbf{F}_i can also contain a special symbol ϵ that indicates a word skip.

probability than the preposition *li*. The second, at position six, misidentifies the preposition *AIY* (to) as *EIY* (on).

2.6.3.3.2 Decoding confusion networks with hierarchical models

Bertoldi *et al.* (2007) demonstrated that phrase-based translation systems can be adapted to efficiently translate ambiguous input represented as a confusion network. Given the promising performance of the hierarchical translation models introduced by Chiang (2007), which enable MT systems to capture hierarchical structure without the necessity of appealing to treebank-derived parsing, we found it natural to extend decoding for such models in a similar way, in order to defer the resolution of preprocessing ambiguity (Dyer 2007a).

Following Shieber *et al.* (1995), Goodman (1999) and Chiang (2007), we characterize hierarchical analysis, i.e., parsing, as a deductive proof system. In an inference rule

$$\frac{I_1:w_1 \cdots I_k:w_k}{I:w} \phi$$

when antecedents $I_1 \cdots I_k$ are true with weights $w_1 \cdots w_k$, then consequent I is provable with weight w . Axioms are inference rules with no antecedents. ϕ is any number of side conditions that must be true for the inference to be made.

Traditional CKY-style parsing of an input sequence $f_1 \cdots f_n$ is captured concisely by inference rules (1) and (2), where item $[X, i, j] : w$ indicates that a subtree of category X has been recognized spanning $f_i \cdots f_j$ with weight w . The classic dynamic programming version of CKY is merely a systematic search for a proof of $[S, 0, n]$ involving memoization of inferred items.

$$\frac{Z \rightarrow f_{i+1} : w}{[Z, i, i+1] : w} \quad (2.53)$$

$$\frac{Z \rightarrow XY : w_1 \quad [X, i, k] : w_2 \quad [Y, k, j] : w_3}{[Z, i, j] : w_1 \times w_2 \times w_3} \quad (2.54)$$

Extending the algorithm to parse a confusion network \mathcal{G} rather than single best preprocessing analysis $f_1 \cdots f_n$ requires two straightforward modifications.³¹ First, inference rule Equation 2.53 is changed to allow generation of multiple neighbor inferences in the search space by “scanning” any token in column \mathbf{F}_i .

$$\frac{Z \rightarrow \mathbf{F}_{i+1,k} : w}{[Z, i, i+1] : w \times p_{i+1,k}} \quad (2.55)$$

Second, to properly handle ϵ - words, we add:

³¹ Using dynamic programming, the resulting algorithm runs in time $O(n^3d)$, where d is the maximum number of preprocessing alternatives in any column.

$$\frac{[X,i,j]:w}{[Z,i,j+1]:w \times p_{j+1,k}} F_{j+1,k} = \epsilon \quad (2.56)$$

We employ standard log-linear modeling for hierarchical translation (Och and Ney 2002; Chiang 2007); the probability $P(D)$ for a synchronous derivation D is modeled as

$$P(D) \propto \prod_i \prod_{I_k \in D} h_i(I_k)^{\lambda_i} \quad (2.57)$$

where the I_k are inferred items $[X, i, j]$ in the derivation, h_i are model features and the λ_i are log-linear parameters tuned on held-out development data.

2.6.3.3.3 Experimental results

Dyer (2007a) established the utility of confusion network input for hierarchical decoding by demonstrating significant performance improvements in Czech-English translation. These experiments used the standard feature set for hierarchical translation modeling (Chiang 2007) and confusion networks were constructed by lemmatizing each word in the test set and having two alternatives at each position in the confusion network: the surface form, and the lemmatized form. Significant improvements over baseline surface and lemma systems were observed. Furthermore, our experiments showed that previous results, indicating that lemmatizing input before translating it yielded better performance, did not generalize to our systems (which used much more training data), suggesting that deferring decisions about the proper level of morphological simplification until decoding time has value.

Another source of ambiguity that is well-suited for representation in a confusion network concerns the proper amount of diacritization that should be used when translating from Arabic. Arabic orthography does not capture all the phonemic distinctions that are made in the spoken language since optional diacritics are used to indicate consonant quality as well as the identity of short vowels. When these diacritics are absent (as is the case in virtually all text genres), a single orthographic form will correspond to several phonemically distinct words. This insight was the motivation for a study by Diab, Ghoneim and Habash, who used SVMs to predict missing diacritics as a preprocessing step for MT (Diab *et al.* 2007). Although a variety of different diacritization schemes were attempted, the study failed to find any set of diacritics that consistently improved translation quality on an Arabic-English task. The authors conclude that the fragmentation of training data resulting from the proliferation of distinctive forms resulted in poorly estimated translation models. However, confusion networks offer the possibility of using the more precise diacritized forms when adequate training data is available but backing off to less exact forms when this is not the case. Table 2.67 shows the results of our Arabic diacritization lattice experiments.³² For each test set, corresponding to the same test sets used by Diab *et al.* (2007), we see

³² For descriptions of the training data, alignment and language models used, refer to the work of Diab *et al.* (2007).

improvements in translation quality as measured by both BLEU and TER.³³ Using bootstrap resampling, all improvements are estimated to be significant at $p < .05$.

Condition	MT03	MT05	MT06
BASELINE (BLEU)	45.0	42.2	44.2
CN (BLEU)	45.9	43.1	45.1
BASELINE (TER)	48.3	47.6	46.5
CN (TER)	47.6	46.7	45.7

Table 2.67: Results of Arabic diacritization confusion network experiments.

2.6.3.4 Preserving Ambiguity Using Lattices

Many forms of preprocessing do not produce alternatives in one-to-one correspondence with the original sequence of input tokens. Chinese word segmentation and Arabic affix splitting are notable examples. It is always possible to approximate more general correspondences using ϵ -transitions. For example, a deferred decision about whether to preserve the surface form of *AlktAb* or to represent it as *Al ktAb* can be represented in a confusion network by using an arc ϵ , so that *AlktAb*+ ϵ has the same span as *Al*+*ktAb*. However, this use of ϵ -transitions creates two problems. First, the resulting confusion network overgenerates; that is, it permits paths that would not be possible if a single arc for *AlktAb* could be placed in direct correspondence with two arcs representing the distinct tokens *Al* and *ktAb*. Second, it can be difficult to decide where the ϵ -transition should be deployed: why *AlktAb*+ ϵ and not ϵ +*AlktAb*?

These problems disappear if alternative analyses are allowed to use any number of arcs necessary to span the same portion of the original input, i.e., if the ambiguous alternatives are represented using a lattice structure. In 2008, Dyer *et al.* introduced decoding of lattices within hierarchical phrase-based modeling frameworks, which we briefly summarize here.

2.6.3.4.1 Definition

A word lattice $\mathcal{G} = \langle V, E \rangle$ is a directed acyclic graph that formally constitutes a weighted finite state automaton (FSA) and we will stipulate that exactly one node has no outgoing edges and is designated the ‘end node’. It is worth noting that because of these restrictions, the FSA is guaranteed to describe a finite set of strings. An edge $e_i \in E$ is labeled with a word $f(e_i)$ (in the case of our model, in the source language) or to indicate an empty transition. Each edge is associated with a cost, which for present purposes we assume is scalar.³⁴

Two special classes of restricted word lattices deserve mention. First, an FSA generating a single sentence is the most basic restricted lattice possible. Second, confusion networks are a special case where the linear structure of a sentence is preserved and alternative arcs between adjacent nodes permit the encoding of a multitude of paths.

³³ TER is translation edit rate so lower scores are better.

³⁴ We have implemented vector-valued edge costs, as well, but do not describe those here.

0			1			2		
F_{0j}	p_{0j}	R_{0j}	$F_{1,j}$	$p_{1,j}$	$R_{1,j}$	$F_{2,j}$	$p_{2,j}$	$R_{2,j}$
a	1	1	b	1	2	c	1	3
a	1/3	1	b	1	2	c	1/2	3
x	1/3	1				d	1/2	3
ϵ	1/3	1						
x	1/2	1	y	1	2	b	1/2	3
a	1/2	2				c	1/2	3

Table 2.68: Topologically ordered chart encoding of the three lattices in Figure 2.50. Each cell ij in this table is a triple $\langle F_{ij}, p_{ij}, R_{ij} \rangle$

For translation, we will find it useful to encode a lattice G in a chart based on a topological ordering of the nodes, as described by Cheppalier *et al.* (1999). The chart representation of the graph is a triple of two dimensional matrices $\langle \mathbf{F}, \mathbf{p}, \mathbf{R} \rangle$, which can be constructed from the topologically numbered graph. $F_{i,j}$ is the word label of the j^{th} transition leaving node i . The corresponding transition cost is $p_{i,j}$. $R_{i,j}$ is the node number of the node on the right side of the j^{th} transition leaving node i . Note that $R_{i,j} > i$ for all i, j . Table 2.68 shows the word lattice from Figure 2.50 represented in matrix form as $\langle \mathbf{F}, \mathbf{p}, \mathbf{R} \rangle$.

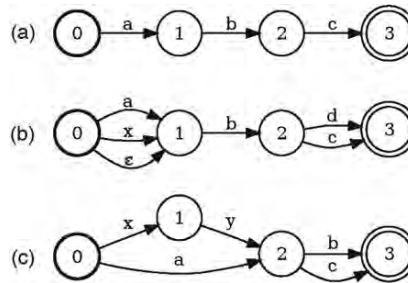


Figure 2.50: Three examples of word lattices: (a) sentence, (b) confusion network and (c) non-linear word lattice. Nodes are numbered according to a topological ordering.

2.6.3.4.2 Decoding lattices with hierarchical models

We again describe hierarchical phrase-based analysis via inference rules in a deductive proof system. Although there are a variety of context-free parsers that can parse a word lattice, we present a bottom-up algorithm for parsing general word lattices with unrestricted (i.e., non-binarized) context-free grammars, since the synchronous grammars learned in typical modeling are not in Chomsky-normal form. The parser we present here is a generalization of a CKY parser for lattices presented by Cheppalier *et al.* (1999).

Axioms:

$$\frac{}{[X \rightarrow \bullet \gamma, i, i]: w} (X \xrightarrow{w} \langle \gamma, \alpha \rangle) \in G, i \in [0, |V| - 2]$$

Inference rules:

$$\frac{[X \rightarrow \alpha \bullet \mathbf{F}_{j,k} \beta, i, j]: w}{[X \rightarrow \alpha \mathbf{F}_{j,k} \bullet \beta, i, \mathbf{R}_{j,k}]: w \times \mathbf{p}_{j,k}}$$

$$\frac{[X \rightarrow \alpha \bullet \beta, i, j]: w}{[X \rightarrow \alpha \bullet \beta, i, \mathbf{R}_{j,k}]: w \times \mathbf{p}_{j,k}} \mathbf{F}_{j,k} = \epsilon$$

$$\frac{[Z \rightarrow \alpha \bullet X \beta, i, k]: w_1 \quad [X \rightarrow \gamma \bullet, k, j]: w_2}{[Z \rightarrow \alpha X \bullet \beta, i, j]: w_1 \times w_2}$$

Goal state:

$$[S \rightarrow \gamma \bullet, 0, |V| - 1]$$

Briefly, the three inference rules, which are shown above are as follows: (1) match a terminal symbol and move across one edge in the lattice; (2) move across an ϵ -edge without advancing the dot in an incomplete rule; (3) advance the dot across a non-terminal symbol given appropriate antecedents. The first of these applies when the next symbol in the grammar rule is a terminal that matches the label on the j^{th} transition out of node i , i.e. a “scan” operation. The second is used to advance across any null transitions in the word lattice. The third rule applies when it has been proven that a non-terminal of type X starting at node i exists, i.e. a “completer” step.

The parser we have just described generalizes a standard sentence parser to accept word lattices. Note that any derivation in the translation forest picks out a distinct path through the source lattice in addition to a translation. For more information on the issues associated with decoding, refer to the work of Dyer *et al.* (2008).

2.6.3.4.3 Experimental results

Both Chinese and Arabic orthography concatenate morphemes that are separated by whitespace in English. This situation results in large vocabularies in these languages which can lead to poor estimations of translation models. To address this, some kind of segmentation preprocessing is necessary. However, as we have indicated, the appropriate level of segmentation may depend on a variety of factors and vary even within a single sentences. Furthermore, the process of generating a segmentation for Chinese is inherently ambiguous: not even native speakers can agree on the appropriate segmentation granularity. Word lattices represent a very compact way of encoding segmentation alternatives and enabling their resolution to be deferred until during decoding. In this section, we report results showing the effect of using segmentation lattices when translating from Arabic and Chinese into English.

In our experiments we used two state-of-the-art Chinese word segmenters: one developed at Harbin Institute of Technology (Zhao *et al.* 2001), and one developed at Stanford University (Tseng *et al.* 2005). In addition, we used a character-based

segmentation. In the remainder of this section, we use *cs* for character segmentation, *hs* for Harbin segmentation and *ss* for Stanford segmentation. We built two types of lattices: one that combines the Harbin and Stanford segmenters (*hs+ss*) and one which uses all three segmentations (*hs+ss+cs*). The systems used in these experiments were trained on the NIST MT06 Eval training corpus, without the UN data. The Chinese portion of the corpus was analyzed with the three segmentation schemes. For the systems using word lattices, the training data contained the versions of the corpus appropriate for the segmentation schemes used in the input. Table 2.69 summarizes the results for Chinese-English translation using a standard hierarchical phrase-based translation model. On both test sets, there were significant gains for using segmentation lattices over any single-best segmentation and as more segmentations were added, the further the quality of the translation improved. Results are reported using case-insensitive BLEU with 4 reference translations.

(Source Type)	Language	MT05	MT06
Cs	Chinese	29.0	28.2
Hs		30.0	29.1
Ss		30.7	29.6
hs+ss		31.3	30.1
hs+ss+cs		31.8	30.4
SURFACE	Arabic	52.5	39.9
MORPH		53.8	41.8
MORPH+SURFACE		54.5	42.9

Table 2.69: Chinese word segmentation and Arabic morpheme segmentation results in BLEU.

Arabic orthography, although it uses spaces to delimit many more morphemes than Chinese, is nevertheless problematic for MT since a large class of functional elements (prepositions, pronouns, tense markers, conjunctions, definiteness markers) are attached to their host stems. Thus, while the training data may provide good evidence for the translation of a particular stem by itself, the same stem may not be attested when attached to a particular conjunction. We, therefore, conjectured that Arabic-English translation might also benefit from deferring segmentation decisions until decoding. To test this, we created lattices from an unsegmented version of the Arabic test data and generated alternative arcs where clitics (mostly prepositions), as well as, the definiteness marker (*Al-*) and the future tense marker (*s-*) were segmented into tokens.³⁵ We used the Buckwalter morphological analyzer and disambiguated the analysis using a simple unigram model trained on the Penn Arabic Treebank. Table 2.69 summarizes the results for lattices versus two baseline systems and we again can see that there are advantages for including multiple segmentation alternatives in a lattice.

³⁵ This corresponds approximately to the D3 segmentation described by Habash and Sadat (2006).

2.6.3.5 Conclusions

By introducing the decoding of source lattices, we have opened up a new avenue of inquiry based on a key insight that is absent from most current MT research: the given source language sentence is not the only way an author's meaning could have been expressed. This is true for low-level preprocessing phenomena of the kind we have discussed so far: if you give several Chinese speakers the same written sentence, they may disagree on the correct word segmentation even if they agree on the meaning! It is also clear for other issues such as morphological analysis. But we would argue that one can take this idea even further.

Consider, for convenience, an example using English as the source: *The Democratic candidates stepped up their attacks during the debate*. Current syntax-based statistical MT systems would decode based on analysis of this sentence's structure, with particular treatment of constituents, dependencies, morphological pre-analysis, etc. varying from approach to approach. However, the same basic meaning could have been expressed in many different ways.

- During the debate the Democratic candidates stepped up their attacks.
- The Democratic contenders ratcheted up their attacks during the debate.
- The Democratic candidates attacked more aggressively during the debate.
- The candidates in the Democratic debate attacked more vigorously.

Some variations are lexical (candidates versus contenders, attacks versus attacked) and others reflect syntactic alternatives for packaging up the combination of attacking and increasing, e.g., whether the attacking or the increasing serves as the main verb. However, the observation underlying the work we have reported here holds at all levels: given the available training data, some parts of the sentence's meaning may be better translated when expressed one way, while other parts may be better translated when expressed another way and we want to employ the power of the entire model in order to make those choices in a fine-grained way.

We plan, therefore, to use source-sentence lattices to further investigate the power of encapsulating source-sentence expression alternatives, now tackling a whole range of variations from those closer to the surface (segmentation, morphology) all the way up to variations at the level of meaning (lexical choice, syntactic alternations, paraphrase). Source lattices are the key to attacking these problems in a unified setting and, crucially, exploiting generalizations about the source language while still maintaining the surface-to-surface orientation, trainability and decoding paradigm that characterizes the statistical state of the art.

2.7 Adaptation and Data Selection

2.7.1. BiTAM: Bilingual Topic AdMixture Models for Bilingual Topic Exploration, Word Alignment and Translation

Authors: Bing Zhao and Eric P. Xing

2.7.1.1 Introduction

Most SMT systems view parallel data as independent sentence-pairs whether or not they are from the same document-pair. As a result, translation models are learned only at sentence-pair level and sentences are translated indifferently, because the document context and structure – for translating documents – are generally ignored. In fact, translating documents differs considerably from translating a group of unrelated sentences. A sentence, when taken out of the context from the document, is generally more ambiguous and less effective for translation. One should avoid destroying a coherent document by simply translating it into a group of sentences which are unrelated to each other and detached from the context.

Developments in statistics, genetics and machine learning have shown that the latent semantic aspects of complex data can often be captured by a model known as the *statistical admixture* (or mixed membership model of Erosheva *et al.* (2004)). Statistically, an object is said to be derived from an admixture if it consists of a bag of elements, each sampled independently or coupled in a certain way, from a mixture model. In the context of SMT, each parallel document-pair is treated as an object. Depending on the chosen modeling granularity, all sentence-pairs or word-pairs in a document-pair correspond to the basic elements constituting the object and the mixture from which the elements are sampled can correspond to a collection of translation lexicons and monolingual word frequencies based on different topics (e.g., economics, politics, sports, etc.). Variants of admixture models have appeared in population genetics, as published by Pritchard *et al.* (2000) and text modeling, as published Blei *et al.* (2003) and Erosheva *et al.* (2004).

Recently, we proposed a Bilingual *Topic-AdMixture* (BiTAM) means (Zhao and Xing 2006) of modeling the topical aspects of SMT; this enables word-pairs from a parallel document-pair to share one topic-specific translation lexicons, with proper document-specific weights. BiTAMs, generalizations over IBM Model-1, are efficient to learn and scalable for large training data. However, they do not capture locality constraints of word alignment, i.e., words “close-in-source” are aligned to words “close-in-target”, under document-specific topical assignments. To incorporate such constraints, we integrate the strengths of both HMM and BiTAM and propose a Hidden Markov Bilingual Topic-AdMixture model, or HM-BiTAM as used by Zhao and Xing (2007), for word alignment to leverage both locality constraints and topical context underlying parallel document-pairs. Related works of Saiz (2008), skipping the Bayesian part of the mixtures priors, showed limited improvements on using mixtures of translation models of IBM Model-1 IBM Model-2 and HMMs. Improvements are also reported on adapting the

language models using topical mixtures defined over bilingual data (Tam *et al.* 2007) and adapting translation models using mixtures with mixture-weights assigned using empirical methods based on empirical text distance defined on data partitions. This section focuses on Bayesian learning to mix the underlying translation models in various hierarchical structures.

In the BiTAM framework, a topic corresponds to a point in a *conditional simplex*: each source word invokes a simplex, in which each dimension corresponds to a bilingual translation candidate. The hidden topics are leveraged in learning topic-specific bilingual translation lexicons, to enrich the emission distributions in IBM models or HMM, and thereby, improve their modeling expressiveness. Under this framework, bilingual statistics are shared more effectively across different topics. Constrained by the hidden topics, a word will have only limited translation candidates. The translation models, therefore, are expected to be *smaller* and *sharper*.

In this work, we start from the simplest BiTAM, drawn upon IBM Model-1, to introduce the basics. We examine HM-BiTAM, integrating HMM, to estimate topic-specific word-to-word translation lexicons (lexical mappings), as well as the monolingual topic-specific word-frequencies for both languages, via variational learning from the document-level context in parallel document-pairs. HM-BiTAM models offer a principled way of inferring optimal translation from a given source language in a context-dependent fashion. We report an extensive empirical analysis of HM-BiTAM, in comparison with the related methods. The results show our model’s effectiveness on the word-alignment task. We also demonstrate two application aspects: the utilization of HM-BiTAM for bilingual topic exploration and its application for improving translation qualities.

2.7.1.2 Notations and Terminologies

To identify the latent variables to capture the abstract notations of bilingual topics, we start from a revisit to the parallel data to introduce the entities in our BiTAM models including “word-pair”, “sentence-pair” and “document-pair” following the notations used in Brown *et al.* (1993):

- A *word-pair* (f_j, e_i) is the basic unit discrete data for machine translation, where f_j is a French word and e_i is an English word. j and i are the positions in the corresponding French sentence f and English sentence e .
- A *sentence-pair* (\mathbf{f}, \mathbf{e}) contains the source sentence f with a length of J : $\mathbf{f} = f_1, f_2, \dots, f_j, \dots, f_J$; a target sentence e with a length of I : $\mathbf{e} = e_1, e_2, \dots, e_j, \dots, e_I$. The two sentences \mathbf{f} and \mathbf{e} are translation of each other.
- A *document-pair* (\mathbf{F}, \mathbf{E}) contains two documents: source document: \mathbf{F} and the target \mathbf{E} , which are mutual translations of each other. \mathbf{F} has N sentences: $\mathbf{F} = \{\mathbf{f}_n | n = 1, \dots, N\}$. For simplicity, let’s assume the sentence-level alignment is one-to-one mapping. Therefore, a document-pair has a sequence of N parallel sentence-pairs $\{(\mathbf{f}_n, \mathbf{e}_n) | n = 1, \dots, N\}$, where $(\mathbf{f}_n, \mathbf{e}_n)$ is the n^{th} parallel sentence-pair.

- A *parallel-corpus* \mathbf{C} is a collection of M parallel document-pairs, indexed by d : $\{(\mathbf{F}, \mathbf{E})_d | d = 1, \dots, M\}$. For notations used in the noisy channel model, we denote the French part monolingual corpus as \mathbf{C}_F and English part as \mathbf{C}_E . We are using the *end-user terminology*: *French* is the *source* language and *English* is the *target* language; we are translating foreign language (e.g., French) into English.

Traditional statistical machine translation uses a noisy-channel model to describe the translating process:

$$\mathbf{e}^* = \operatorname{argmax}_{\{\mathbf{e}\}} P(\mathbf{f}|\mathbf{e})P(\mathbf{e}) \quad (2.58)$$

where \mathbf{e} is a English sentence, decoded as the translation of the source sentence \mathbf{f} ; $P(\mathbf{f}|\mathbf{e})$ is the *translation model*, for example, the traditional IBM Model 1 ~ 5 and HMM; $P(\mathbf{e})$ is a *language model*, such as a trigram.

Now, the translation model can be extended to the document-level using the notations given in the above. Instead of translating at the sentence-level, we are able to translate at the document-level, shown in Eqn. 2.59.

$$\mathbf{C}_E^* = \operatorname{argmax}_{\{\mathbf{C}_E\}} P(\mathbf{C}_F|\mathbf{C}_E)P(\mathbf{C}_E) \quad (2.59)$$

where $P(\mathbf{C}_F|\mathbf{C}_E)$ is a document-level translation model: a generative model for the whole document of \mathbf{C}_F as one entity. In this model, we are able to introduce the topics for each document-pair to improve the model's expressive power. Rather than being translated into unrelated segments and then pieced together later on, as presented in traditional approaches, the sentences in the new model can meaningfully be threaded and integrated during translation by the topics in the documents. As a result, the translation is more coherent since the document is treated as a whole entity.

A document-pair (\mathbf{F}, \mathbf{E}) is treated as an admixture of topics, which is induced by random draws of a topic, from a pool of topics, for each sentence-pair. A unique normalized and real-valued vector θ , referred to as a *topic-weight vector*, captures the contribution of different topics and is instantiated for each document-pair, so that the sentence-pairs with their alignments are generated from topics mixed according to these common proportions. Marginally, a sentence-pair is word aligned according to a unique topic-specific bilingual model given the hidden topical assignments. Therefore, the sentence-level translations are coupled, rather than being independent as assumed in the IBM or HMM models and their extensions.

We will introduce the proposed document-level translation models, starting from extending IBM Model-1 to BiTAM Model-1 and then moving to HMM, with similar derivations, to finalize HM-BiTAM.

2.7.1.3 Bilingual Topic Admixture Model-1

The first model BiTAM-1, proposed within this BiTAM framework, generalizes over the simplest IBM Model-1 style word alignment. BiTAM-1 is a generative model: each parallel document-pair is represented as a random mixture over a set of latent topics, in which each topic is characterized by a distribution over aligned word pairs.

2.7.1.3.1 Sampling Scheme for BiTAM Model-1

The generative process (i.e., sampling process) for a parallel document-pair (\mathbf{F} , \mathbf{E}) is described as below:

1. Sample number of sentence-pairs N from a Poisson(γ).
2. Sample topic assignment θ_d from a Dirichlet (α).
3. For each sentence-pair (\mathbf{f} , \mathbf{e}) in the document,
 - (a) Sample source sentence length J from a Poisson(δ).
 - (b) Sample a topic z_j from Multinomial(θ_d);
 - (c) Sample a word alignment link a_j from a Dirichlet (ζ).
 - (d) Sample a foreign word f_j according to $p(f_j|\mathbf{e}, a_j, z_j, \beta_{z_j})$.

The parameters of this model are: K topic indicator variables for each parallel document-pair: t_1, t_2, \dots, t_K ; A K -dimensional Dirichlet random variable θ_d , which takes values in the $(K-1)$ -simplex and has the following probability density on this simplex:

$$p(\theta, \alpha) = \frac{\Gamma(\sum_{k=1}^K \alpha_k)}{\prod_{k=1}^K \Gamma(\alpha_k)} \theta_1^{\alpha_1-1} \dots \theta_K^{\alpha_K-1} \quad (2.60)$$

where the parameter α is a K -dim vector with each component $\alpha_k > 0$ and $\Gamma(x)$ is the Gamma function. The alignment vector is $\mathbf{a}=\{a_1, a_2, \dots, a_J\}$, with $a_j=i$ maps the French word f_j to the English word e_i . To be more specific, word alignment link a_j is a selector, which selects the English word at position a_j to be aligned to the French word f_j at the position j in the French sentence. The word-level topic-specific translation lexicon probabilities are parameterized by a $k \times |V_F| \times |V_E|$ table: β , where $|V_E|$ is the size of English vocabulary and $|V_F|$ is the size of French vocabulary. $\beta_{k,e,f} = p(f = f_j | e = e_i, z = k)$ is a corpus-level parameter: topic-specific translation lexicon. In practice, this table is very sparse: one French word f has on average a few candidate English word e translations given the topic assignment of z ; this table is usually much smaller than the standard IBM Model-1: $p(f|e)$, which does not consider topical context at all.

The last two sub-steps in the sampling scheme define a translation model: an *alignment link* a_j is proposed and an observation of f_j is generated according to the proposed topic-specific distributions and the word alignment. In this BiTAM-1, the generative scheme is simplified starting from as simple as the one in IBM Model-1: a_j is sampled independently and uniformly.

The number of sentence-pairs N is independent of the other data generating variables θ , z and a . Thus, its randomness is generally ignored in our modeling. The same assumption applies to the variable of J from a Poisson(δ) and we choose the same uniform distribution as used in IBM models and ignore the prior for a_j from a Dirichlet (ζ) in this section. Also, we do not consider the modeling of \mathbf{E} in the translation model within the noisy-channel paradigm. The *graphical model* describing this generative model of BiTAM-1 is shown in Figure 2.51.

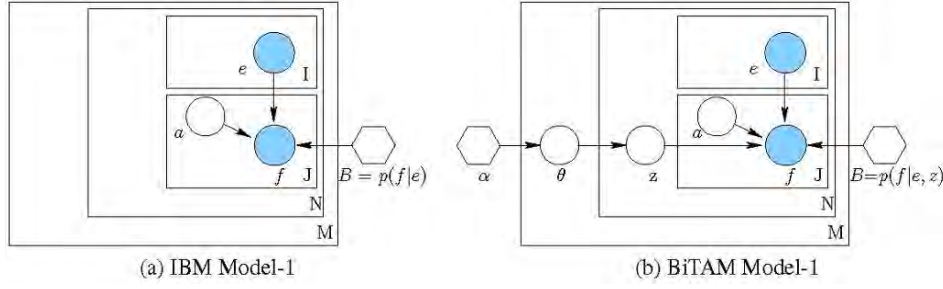


Figure 2.51: Graphical Model representations for (a) IBM Model-1 and (b) BiTAM Model-1. Model parameter for IBM Model-1 in figure (a) is $B = p(f|e)$, which is a simple word-to-word translation lexicon; Model parameters for BiTAM Model-1 (in (b)) are topic-specific lexicons: $B = p(f|e, z)$ and a Dirichlet prior parameter α . All the plates represent replicates. The outmost plate (M -plate) represents M bilingual document-pairs, while the inner N -plate represents the N repeated choice of topics for each sentence-pairs in the document; and the inner J -plate represents J word-pairs within each sentence-pair. Shaded nodes are observations; unshaded nodes are hidden variables. α and β are the corpus level parameters. B , for BiTAM, is a three-dimensional matrix, representing the topic-specific translation lexicons: $p(f|e, z)$.

Note, for BiTAM models within one parallel document-pair, the sentence-pairs are connected by the hidden node θ_d . Therefore, the sentence-pairs are no longer independent of one another as in the traditional IBM Models; they are conditionally independent given the topic mixture assignment of θ_d . In this model, the simplified assumption is that each sentence-pair has a topic. Therefore, the word-pairs within a sentence-pair are not independent as in the traditional IBM Model-1; they are conditionally independent given the sentence-pair's topic z .

Given the parameters of α , β and a set of N alignment vectors $\mathbf{A} = \{\mathbf{a}_n | n = 1, \dots, N\}$, the conditional distribution of a topic mixture θ , a set of N topics \mathbf{z} and a set of N bag-of-word observations \mathbf{f} is given by:

$$p(\mathbf{F}, \mathbf{A}, \theta, \mathbf{z} | \mathbf{E}, \alpha, \beta) = p(\theta | \alpha) \prod_{n=1}^N p(z_n) P(\mathbf{f}_n, \mathbf{a}_n | \mathbf{e}_n, \beta_{z_n}),$$

where $p(z_n | \theta) = \theta_i$ is the probability for sampling a topic z , such that $z_n^i = 1$, where i is a unique topic index. Marginalizing out θ and z , we can obtain the marginal distribution of generating \mathbf{F} from \mathbf{E} for each parallel document-pair, as shown below:

$$p(\mathbf{F}, \mathbf{A} | \mathbf{E}, \alpha, \beta) = \int p(\theta | \alpha) \left(\prod_{n=1}^N \sum_{z_n} p(z_n | \theta) p(\mathbf{f}_n, \mathbf{a}_n | \mathbf{e}_n, \beta_{z_n}) \right) d\theta, \quad (2.61)$$

where $p(\mathbf{f}_n | \mathbf{e}_n, \mathbf{a}_n, \beta_{z_n})$ is a topic-specific sentence-level translation model under the topic assignment of z_n . After marginalizing out the hidden topic assignments z , we have:

$$p(\mathbf{f}_n | \mathbf{e}_n, \mathbf{a}_n, \theta) = \sum_{z_n} p(z_n | \theta) p(\mathbf{f}_n | \mathbf{e}_n, \mathbf{a}_n, \beta_{z_n}), \quad (2.62)$$

where $p(z_n | \theta) = \theta_i$ is topic weight for choosing topic- i and this reveals that the proposed sentence-level alignment model is, in essence, an *additive mixture* of topic-specific translation models.

According to the simplified model shown in Figure 2.51, the French words f_j 's are conditionally independent of each other, the alignment variables a_j 's are independent of

other variables and further more we assume all alignments are equally probable for simplicity reason. Now the probability distribution for each sentence-pair is further simplified:

$$p(\mathbf{f}_n, \mathbf{a}_n | \mathbf{e}_n, \beta_{z_n}) \approx \frac{1}{J} \prod_{j=1}^J p(f_j | e_{a_j}, \beta_{z_n}). \quad (2.63)$$

The translation model for the whole parallel corpus is given by taking the product of the marginal probabilities of each single document as shown in the following equation:

$$p(\mathbf{C}_F | \mathbf{C}_E, \alpha, \beta) \propto \prod_{d=1}^M \int p(\theta_d | \alpha) \sum_{\{\mathbf{C}_A\}} \left(\prod_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn} | \theta_d) \prod_{j=1}^{J_{dn}} p(\mathbf{f}_{dn}, \mathbf{a}_{dn} | \mathbf{e}_{dn}, \beta_{z_{dn}}) \right) d\theta_n \quad (2.64)$$

Overall, in this generative model, the model parameters are at different levels. The parameters of α and β are corpus-level parameters; θ_d is the document-level parameter, sampled once per document-pair; the variable $\beta_{z_{dn}}$ is the sentence-level parameter, sampled once per sentence-pair in the document-pair; the variables of f_{dnj} and a_{dnj} are word-pair level variables, sampled for each word-pair in the sentence-pair.

2.7.1.3.2 Inference and Learning for BiTAM Model-1

For inference and learning, the posterior distribution of the hidden variables given a document-pair is:

$$p(\mathbf{A}, \theta, \mathbf{z} | \mathbf{F}, \mathbf{E}, \alpha, \beta) = \frac{p(\mathbf{F}, \mathbf{A}, \theta, \mathbf{z} | \mathbf{E}, \alpha, \beta)}{p(\mathbf{F} | \mathbf{E}, \alpha, \beta)}. \quad (2.65)$$

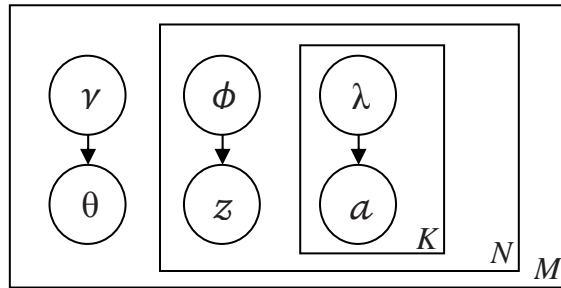


Figure 2.52: Graphical Model representation of the variational distribution to approximate the posteriors in Bilingual AdMixture Model-1.

Due to the hybrid nature of the “V” structure (explain-away structure) in the graphical model in Figure 2.51, the hidden variables of \mathbf{A} , θ and \mathbf{z} are all coupled together. This makes the joint posterior distribution intractable to compute. We resort to the generalized mean field approximation as used by Xing *et al.* (2003) and carry out the variational inference. The variational inference is essentially to use Jensen’s inequality to obtain an adjustable lower bound on the log-likelihood, which is easier to optimize.

Shown in Figure 2.52, a simple way to obtain a tractable family of lower bounds is to break the graphical model into smaller isolated pieces and decouple the estimation of the posteriors with additional free *variational parameters* introduced. The family is indexed by the following variational distributions. For a given document-pair, the approximated posterior inference is shown below.

$$q(\theta, \mathbf{z}, \mathbf{a}) = q(\theta|\gamma) \prod_{n=1}^N q(z_n|\phi_n) \prod_{n=1}^N q(a_{nj}|\lambda_{nj}) \quad (2.66)$$

where the Dirichlet parameter γ and the multinomial parameters (ϕ_1, \dots, ϕ_n) and the conditional multinomial parameters $(\lambda_{n1}, \dots, \lambda_{nJ_n})$ are the free parameters. These parameters are document specific.

The estimation of the parameters is *variational EM*. In the E-step, for each document-pair, find the optimal values for the variational parameters; in the M-step, maximize the resulting lower bound on the log-likelihood with respect to the model parameters of α and β .

For machine translation, the inference algorithm we want should be fast and efficient. We use variational EM with deterministic annealing (Smith and Eisner 2004) to overcome the local optimum.

According to the assumption of *exchangeability* for document-pairs, the overall log likelihood of a parallel corpus is the sum of the log likelihood of each individual document-pair; moreover, the overall variational lower bound is the sum of each individual variational bounds. The updating equations for the variational parameters are straightforward to compute via a fix-point algorithm, as listed as below:

$$\gamma_k = \alpha_k + \sum_{n=1}^{N_d} \phi_{dnk} \quad (2.67)$$

$$\phi_{dnk} \propto \exp(\Psi(\gamma_k) - \Psi[\sum_{k'}^K \gamma_{k'}]) \exp\left(\sum_j^{J^{dn}} \sum_i^{I^{dn}} \phi_{dnji} \log \beta_{k,e_i,f_j}\right) \quad (2.68)$$

$$\lambda_{dnji} \propto \exp\left(\sum_{k=1}^K \phi_{dnk} \log \beta_{k,e_i,f_j}\right) \quad (2.69)$$

To update the topic-specific lexicons $\beta_{k,e,f}$, we have:

$$\beta_{k,e,f} \propto \sum_d^M \sum_{n=1}^{N_d} \sum_{j=1}^{J^{dn}} \sum_{i=1}^{I^{dn}} \phi_{dnk} \delta(f, f_j) \delta(e, e_i) \lambda_{dnji} \quad (2.70)$$

The true prior of the observed document-pair is unknown, but it can be learned iteratively. To learn the parameter of Dirichlet prior α , one can use gradient ascent approach (Sjölander *et al.* 1996). At each iteration, $\alpha_k > 0$ is updated using the sufficient statistics of γ_k collected from the whole corpus.

2.7.1.4 Hm-BiTAM: Hidden Markov Bilingual Topic AdMixture Model

In this section, we generalize over HMM, a key component for most of the state-of-the-art SMT systems, within BiTAM framework, using similar derivations as described in the previous section for BiTAM Model-1, to formulate Hidden Markov Bilingual Topic AdMixture (HM-BiTAM) models.

2.7.1.4.1 Hidden Markov Model for Sentence-Pairs

Within a sentence-pair, Vogel *et al.* (1996) implement the “proximity-bias” assumption, that words “close-in-source” are aligned to words “close-in-target”, which is effective for improving word alignment accuracies, especially for linguistically close language-pairs.

A graphical model representation for such an HMM is illustrated in Figure 2.53(a). Specifically, we introduce the mapping $j \rightarrow a_j$, which assigns a French word f_j in position j to an English word e_i in position $i = a_j$ denoted as e_{a_j} . Each (ordered) French word f_j is an observation and it is generated by an HMM state defined as $[e_{a_j}, a_j]$, where the alignment indicator a_j for position j is considered to have a dependency on the previous alignment a_{j-1} . Thus, a first-order HMM for an alignment between $e \equiv e_{1:I}$ and $f \equiv f_{1:J}$ is defined as:

$$p(f_{1:J}|e_{1:I}) = \sum_{a_{1:j}} \prod_{j=1}^J p(f_j | e_{a_j}) p(f_{1:J}|e_{1:I})$$

where $p(a_j|a_{j-1})$ is the *state transition probability*; J and I are *sentence lengths* of the French and English sentences, respectively. The transition model enforces the proximity bias.

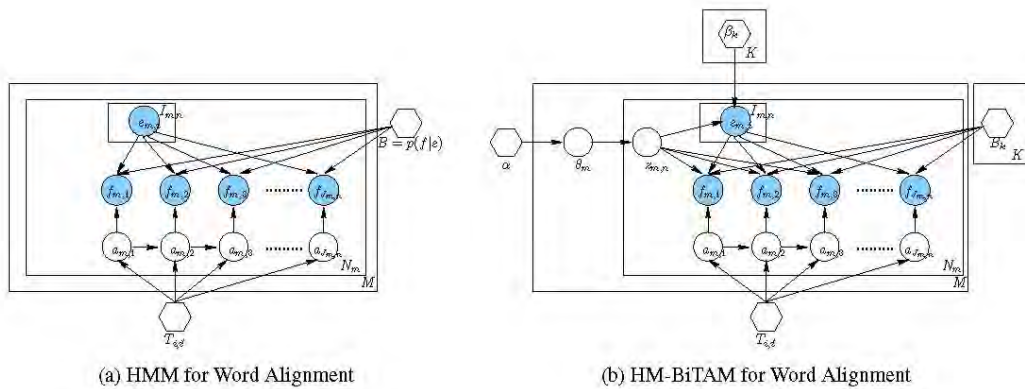


Figure 2.53: The graphical model representations of (a) HMM and (b) HM-BiTAM, for parallel corpora. Circles represent random variables, hexagons denote parameters and observed variables are shaded.

Much of HMM’s success relies on the transition probability of $T_{i,i'} = p(a_j = i | a_{j-1} = i')$ which considers only the positions of the source and target tokens. The emission probability $p(f_j | e_{a_j})$, however, totally ignores the lexical context either from sentence-pairs, or document-pairs. Overall, HMM is less sensitive in training toward a better lexicon due to the ignorance to the adjacent lexical context. Systems rely on higher-order IBM models to refine the lexical choices.

In HM-BITAM, we not only incorporate the transition probability of HMM, but also introduce an emission probability which is sensitive to the lexical context: $B_k = p(f|e, k)$,

as shown in the graphical model representation in Figure 2.53(b). The lexical context is abstracted in the choice of a topical mixture k , which is inferred via variation learning with a Dirichlet prior. Each observation f is generated by a vector and each weighted dimension is defined by a tuple: (e, k) , inferred from the context as in BiTAM-1. Given a document-pair (\mathbf{F}, \mathbf{E}) containing N one-to-one sentence-level aligned pairs $(\mathbf{e}_n, \mathbf{f}_n)$, HM-BiTAM implements the following generative scheme.

2.7.1.4.2 A Generative Scheme for HM-BiTAM

Similar to BiTAM-1, with a conjugate prior $\text{Dirichlet}(\alpha)$, the *topic-weight vector* (TWV), θ_m for each document-pair $(\mathbf{F}_m, \mathbf{E}_m)$, is sampled *independently* from each other. Let the non-underscripted θ denote the TWV of a typical document-pair (\mathbf{F}, \mathbf{E}) , a collection of topic-specific translation lexicons be $\mathbf{B} \equiv \{B_k\}$, where $\mathbf{B}_{i,j,k} = P(f = f_j | e = e_i; z = k)$ is the conditional probability of translating e into f under a given topic indexed by z ; the topic-specific monolingual model $\beta \equiv \{\beta_k\}$, which can be the usual LDA-style monolingual unigrams. The sentence-pairs $\{\mathbf{f}_n, \mathbf{e}_n\}$ are drawn independently from a mixture of topics. Specifically (as illustrated also in Figure 2.53(b)):

1. $\theta \sim \text{Dirichlet}(\alpha)$
2. For each sentence-pair $(\mathbf{f}_n, \mathbf{e}_n)$,
 - (a) $z_n \sim \text{Multinomial}(\theta)$ sample the topic
 - (b) $e_{n,1:l_n} | z_n \sim P(e_n | z_n; \beta)$ sample all English words from a monolingual topic model (e.g., a monogram model),
 - (c) For each position $j_n = 1, \dots, J_n$ in \mathbf{f}_n
 - i. $a_{j_n} \sim P(a_{j_n} | a_{j_n-1}; T)$ sample an alignment link a_{j_n} from a first-order Markov process,
 - ii. $f_{j_n} \sim P(f_{j_n} | e_n, a_{j_n}, z_n; \mathbf{B})$ sample a foreign word f_{j_n} according to a topic specific translation lexicon.

From the sampling scheme above, a sentence-pair consists of a mixture of latent bilingual topics; words within the sentence-pair shares the same set of topics; each topic (a point in a *conditional simplex*) is associated with a distribution over bilingual word-pairs. Overall in HM-BiTAM, a word f is generated by two *hidden* factors: a latent topic z drawn from a document-specific distribution over K topics and the English word e identified by the hidden alignment variable a , as summarized in $P(f_{j_n} | e_n, a_{j_n}, z_n; \mathbf{B})$. Both hidden factors will be inferred in a mean-field approximation setup as detailed in the following section.

2.7.1.5 Learning and Inference for HM-BiTAM

The graphical model in Figure 2.53(b) and the sampling scheme in Section 2.7.1.4 describe the conditional dependencies among the variables. For models like HM-BiTAM, one has to resort to approximate methods to exploit conditional independence

relationship for estimating posteriors, which are closely related to *mean-field* methods in statistical physics. A generalized mean-field approximation scheme is presented here for inferring latent variables in HM-BiTAM and a variational EM algorithm for estimating the model parameters. All the algorithms could be easily extended to various HM-BiTAMs where a topic is sampled at a deeper level for each word-pair, or a higher level at document-pair. Please refer to the work of Zhao (2007) for detailed derivations.

2.7.1.5.1 Variational Inference

The complete likelihood of a document-pair (\mathbf{F}, \mathbf{E}) under HM-BiTAM can be expressed as follows:

$$P(\mathbf{F}, \mathbf{E}, \theta, \vec{z}, \vec{a} | \alpha, \beta, T, \mathbf{B}) = p(\theta | \alpha) P(\vec{z} | \theta) P(\vec{a} | T) P(\mathbf{F} | \vec{a}, \vec{z}, \mathbf{E}, \mathbf{B}) P(\mathbf{E} | \vec{z}, \beta), \quad (2.71)$$

where $P(\vec{a} | T) = \prod_{n=1}^N \prod_{j=1}^{J_n} P(a_{j_n} | a_{j_n-1}; T)$ represents the probability of a sequence of alignment jumps; $P(\mathbf{F} | \vec{a}, \vec{z}, \mathbf{E}, \mathbf{B}) = \prod_{n=1}^N \prod_{j=1}^{J_n} P(f_{j_n} | a_{j_n}, e_n, z_n, \mathbf{B})$ is the *document-level* translation probability; and $P(\mathbf{E} | \vec{z}, \beta)$ is the topic-conditional likelihood of the English document based on a topic-dependent monogram model as used in LDA.

To approximate the posterior $p(\vec{a}, \theta, \vec{z} | \mathbf{F}, \mathbf{E})$ we employ a generalized mean field approach and adopt the following factored approximation to the true posterior: $q(\theta, \vec{z}, \vec{a}) = q(\theta | \vec{\gamma}) q(\vec{z} | \vec{\phi}) q(\vec{a} | \vec{\lambda})$, where $q(\theta | \vec{\gamma})$, $q(\vec{z} | \vec{\phi})$ and $q(\vec{a} | \vec{\lambda})$ are reparameterized Dirichlet, multinomial and HMM, respectively, determined by some *variational parameters* that correspond to the expected sufficient statistics of the dependent variables of each factor as shown by Xing *et al.* (2003).

As well known in the variational inference literature, solutions to the above variational parameters can be obtained by minimizing the Kullback-Leibler divergence between $q(\theta, \vec{z}, \vec{a})$ and $p(\theta, \vec{z}, \vec{a} | \mathbf{F}, \mathbf{E})$, or equivalently, by optimizing the lower bound of the expected (over $q(\cdot)$) log likelihood defined by Equation 2.68, via a fixed-point iteration. Due to space limit, we forego a detailed derivation and directly give the fixed-point equations below:

$$\hat{\gamma}_k = \alpha_k + \sum_{n=1}^N \phi_{n,k}, \quad (2.72)$$

$$\begin{aligned} \hat{\phi}_{n,k} \propto & \exp\left(\Psi(\gamma_k) - \Psi\left[\sum_{k=1}^K \gamma_k\right]\right) \times \exp\left(\sum_{i=1}^{I_n} \sum_{j=1}^{J_n} \lambda_{n,j,i} \log \beta_{k,e_{i_n}}\right) \times \\ & \exp\left(\sum_{j,i=1}^{J_n, I_n} \sum_{f \in V_F} \sum_{e \in V_E} \mathbf{1}(f_{j_n}, f) \mathbf{1}(e_{j_n}, e) \lambda_{n,j,i} \log B_{f,e,k}\right), \end{aligned} \quad (2.73)$$

$$\begin{aligned} \hat{\lambda}_{n,j,i} \propto & \exp\left(\sum_{i'=1}^{I_n} \lambda_{n,j-1,i'} \log T_{i,i'}\right) \times \exp\left(\sum_{i'=1}^{I_n} \lambda_{n,j+1,i''} \log T_{i'',i}\right) \times \\ & \exp\left(\sum_{f \in V_F} \sum_{e \in V_E} \mathbf{1}(f_{j_n}, f) \mathbf{1}(e_{i_n}, e) \sum_{k=1}^K \phi_{n,k} \log B_{f,e,k}\right) \times \\ & \exp\left(\sum_{k=1}^K \phi_{n,k} \log B_{k,e_{i_n}}\right) \end{aligned} \quad (2.74)$$

where $\mathbf{1}(\cdot, \cdot)$ denotes an indicator function and $\Psi(\cdot)$ represents the digamma function.

The vector $\hat{\phi}_n \equiv (\hat{\phi}_{n,1}, \dots, \hat{\phi}_{n,K})$ given by Equation 2.74 represents the approximate posterior of the topic weights for each sentence-pair $(\mathbf{f}_n, \mathbf{e}_n)$. The topical information for updating $\hat{\phi}_n$ is collected from three aspects: aligned word-pairs weighted by the corresponding topic-specific translation lexicon probabilities, topical distributions of monolingual English language model and the smoothing factors from the topic prior.

Equation 2.75 gives the approximate posterior probability for alignment between the j -th word in \mathbf{f}_n and the i -th word in \mathbf{e}_n . Intuitively, the first two terms represent the messages corresponding to the *forward* and *backward* passes in HMM; the third term represents the *emission* probabilities and it can be viewed as a geometric interpolation of the strengths of individual topic-specific lexicons; and the last term provides further smoothing from monolingual topic-specific aspects.

Inference of optimum word-alignment One of the translation model's goals is to infer the optimum word alignment: $a^* = \operatorname{argmax}_a P(a|\mathbf{F}, \mathbf{E})$. The variational inference scheme described above leads to an *approximate* alignment posterior $q(\vec{a}, \vec{\lambda})$, which is in fact a reparameterized HMM. Thus, extracting the optimum alignment amounts to applying an Viterbi algorithm on $q(\vec{a}, \vec{\lambda})$.

2.7.1.5.2 Variational EM for parameter estimation

To estimate the HM-BiTAM parameters, which include the Dirichlet hyperparameter α , the transition matrix T , the topic-specific monolingual English monogram $\{\bar{\beta}\}$ and the topic-specific translation lexicon $\{B_k\}$, we employ an variational EM algorithm which iterates between computing variational distribution of the hidden variables (the **E-step**) as described in the previous subsection and optimizing the parameters with respect to the variational likelihood (the **M-step**). Here are the update equations for the M-step:

$$\hat{T}_{i',i'} \propto \sum_{n=1}^N \sum_{j=1}^{J_n} \lambda_{n,j,i'} \lambda_{n,j-1,i'}, \quad (2.75)$$

$$B_{f,e,k} \propto \sum_{n=1}^N \sum_{j=1}^{J_n} \sum_{i=1}^{I_n} \sum_{k=1}^K \mathbf{1}(f_{j_n}, f) \mathbf{1}(e_{i_n}, e) \lambda_{n,j,i} \phi_{n,k}, \quad (2.76)$$

$$\beta_{k,e} \propto \sum_{n=1}^N \sum_{i=1}^{I_n} \sum_{j=1}^{J_n} \mathbf{1}_{e_{i_n}, e} \lambda_{n,j,i} \phi_{n,k}. \quad (2.77)$$

For updating Dirichlet hyperparameter α , which is a corpus-level parameter, we resort to gradient ascent as used by Sjölander *et al.* (1996). The overall computation complexity of the model is linear to the number of topics.

2.7.1.5.3 Exploring Bilingual Topics using HM-BiTAM

HM-BiTAM can be used as a bilingual topic explorer in the LDA-style and beyond. Given paired documents, it can extract each topic in both languages in a consistent fashion, which is not guaranteed otherwise when topics are extracted independently from each language using two separate LDAs.

The topics of English and the foreign language will share similar semantic meanings because of the parallel nature of the training data for machine translation. Shown in Figure 2.53(b), both the English and foreign topics are sampled from the same distribution θ , which is a document-specific, topic-weight vector.

Although there is an inherent asymmetry in the bilingual topic representation in HM-BiTAM (that the mono-lingual topic representations β are only defined on English and the foreign topic representations are implicit via the topical translation models), it is not difficult to retrieve the monolingual topic representations of the foreign language via a marginalization over hidden word alignment. For example, the frequency (i.e., monogram) of foreign word f_w under topic k can be computed by

$$P(f_w|k) \sim \sum_{w'} P(f_w|e_{w'}, B_k) P(e_{w'}|\beta_k) \quad (2.78)$$

As a result, HM-BiTAM can actually be used as a bilingual topic explorer in the LDA-style and beyond. Given paired documents, it can extract the representations of each topic in both languages in a consistent fashion (which is not guaranteed if topics are extracted separately from each language using, e.g., LDA), as well as the lexical mappings under each topics, based on a maximal likelihood or Bayesian principle. In Section 2.7.1.5, we demonstrate outcomes of this application.

2.7.1.6 Experiments

We investigate *three* main aspects of the HM-BiTAM model: word alignment, bilingual topic exploration and machine translation. The Chinese-to-English data was used for experiments. We implemented BiTAMs based on the implementations of IBM Models and HMM in GIZA++ (Och and Ney 2004), which includes refinements such as special a treatment of the jump to a NULL word.

Train	#Doc.	#Sent.	# tokens	
			English	Chinese
TreeBank	316	4172	133,598	105,331
Sinorama04	6367	282176	10,321,061	10,027,095
Sinorama02	2373	103252	3,810,664	3,146,014
Chnews.2005	1001	10317	326,347	270,274
FBIS.BEIJING	6111	99396	4,199,030	3,527,786
XinHua.NewsStory	17260	98444	3,807,884	3,915,267
Parallel translation	33,428	597,757	22,598,584	20,991,767

Table 2.70: Training data statistics

Our training data is a collection of parallel *document-pairs*, with document boundaries explicitly given. As shown in Table 2.70, the corpora used are mainly general newswire, covering various topics, such as *economics*, *politics*, *educations* and *sports*. For word alignment evaluation, our test set consists of 95 document-pairs, with 627 manually-aligned sentence-pairs and 14,769 alignment links in total, from NIST MTEval'01 dry-run data. Word segmentations and tokenizations were also fixed manually for optimal word-alignment decisions. This test set contains relatively long

sentence-pairs, with an average sentence length of 40.67 tokens. The long sentences introduce more ambiguities for alignment tasks.

For testing translation quality, NIST MTEval'02 MT evaluation data is used as development data and ten documents from NIST MTEval'04 MT evaluation are used as the unseen test data. BLEU scores are reported to evaluate translation quality with HM-BiTAM models. More experimental setups for several BiTAM conditions are documented by Zhao (2007).

2.7.1.6.1 Empirical Validation

Word Alignment Accuracy We trained HM-BiATM using parallel corpora of sizes ranging from 6M to 22.6M words and we used the F-measure, the harmonic mean of precision and recall, to evaluate word alignment accuracy. The baseline IBM models were trained using an $1^8h^54^3$ scheme.³⁶ Refined alignments are obtained from both directions of baseline models in the same way as described by Och and Ney (2004).

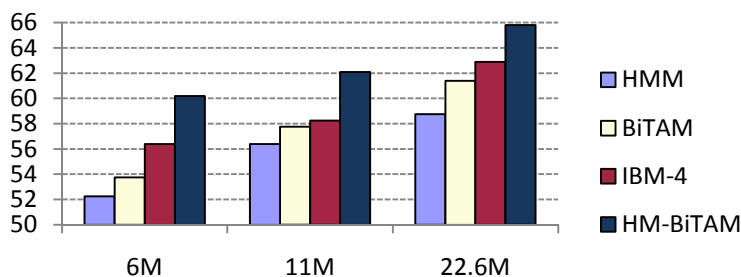


Figure 2.54: Alignment accuracy (F-measure) of different models trained on corpora of different sizes.

Figure 2.54 shows the alignment accuracies of HM-BiTAM, in comparison with that of the baseline-HMM, the baseline BiTAM and the IBM Model-4. Overall, HM-BiTAM gives significantly better F-measures over HMM, with absolute margins of 7.56%, 5.72% and 6.91% on training sizes of 6M, 11M and 22.6M words, respectively. In HM-BiTAM, two key factors contribute to narrowing down the word-alignment decisions: the position and the lexical mapping. The position part is the same as the baseline HMM, implementing the “proximity-bias”. Whereas the emission lexical probability is different, each state is a mixture of topic-specific translation lexicons, of which the weights are inferred using document contexts. The topic-specific translation lexicons are sharper and smaller than the global one used in HMM. Thus, the improvements of HM-BiTAM over HMM essentially resulted from the topic-admixture lexicons adjusted to a particular document-pair. Not surprisingly, HM-BiTAM also outperforms the baseline BiTAM significantly, because BiTAM captures only the topical aspects and ignores the proximity bias.

Notably, HM-BiTAM outperforms IBM Model-4 by a margin of 3.43%, 3.64% and 2.73%, respectively. IBM Model-4 already integrates the fertility and distortion

³⁶ Eight iterations for IBM Model-1, five iterations for HMM and three iterations for IBM Model-4 with deficient EM.

submodels on top of a HMM, which already narrows down the word-alignment choices. However, IBM Model-4 does not have a scheme to adjust its lexicon probabilities specific to document topical-context as in HM-BiTAM. HM-BiTAM wins over IBM-4 by leveraging topic models that capture the document context.

Likelihood on Training and Unseen Documents:

Figure 2.55 shows comparisons of the likelihoods of document-pairs in the training set under HM-BiTAM with those under IBM Model-4 or HMM. Each point in the figure represents one document-pair; the y -coordinate corresponds to the negative log-likelihood under HM-BiTAM and the x -coordinate gives the counterparts under IBM Model-4 or HMM. Overall the likelihoods under HM-BiTAM are significantly better than those under HMM and IBM Model-4, revealing the better modeling power of HM-BiTAM.

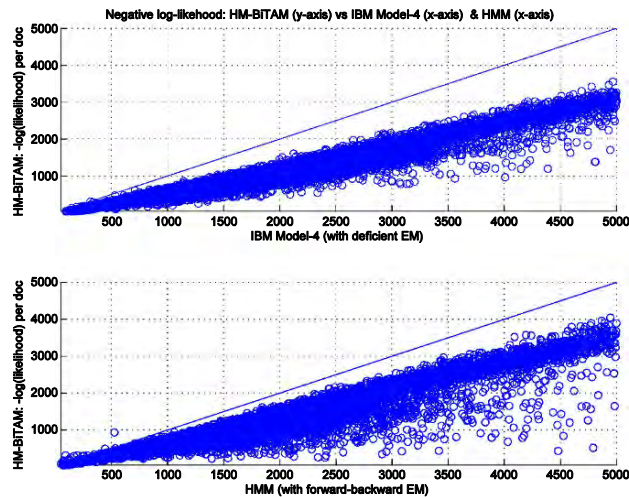


Figure 2.55: Comparison of likelihoods of data under different models. Top: HM-BiTAM vs. IBM Model-4; bottom: HM-BiTAM vs. HMM.

We applied HM-BiTAM to ten document-pairs selected from MT04, which were not included in the training. These unseen document-pairs contain long sentences and diverse topics. As shown in Table 2.71, the likelihoods of HM-BiTAM on these unseen data dominates significantly over HMM, BiTAM and IBM Models in every case, confirming that HM-BiTAM indeed offers a better fit and generalizability for the bilingual document-pairs.

Publisher	Genre	IBM-1	HMM	IBM-4	BiTAM	HM-BiTAM
AgenceFrance(AFP)	news	-3752.94	-3388.72	-3448.28	-3602.28	-3188.90
AgenceFrance(AFP)	news	-3341.69	-2899.93	-3005.80	-3139.95	-2595.72
AgenceFrance(AFP)	news	-2527.32	-2124.75	-2161.31	-2323.11	-2063.69
ForeignMinistryPRC	speech	-2313.28	-1913.29	-1963.24	-2144.12	-1669.22
HongKongNews	speech	-2198.13	-1822.25	-1890.81	-2035	-1423.84
People's Daily	editorial	-2485.08	-2094.90	-2184.23	-2377.1	-1867.13
United Nation	speech	-2134.34	-1755.11	-1821.29	-1949.39	-1431.16
XinHua News	news	-2425.09	-2030.57	-2114.39	-2192.9	-1991.31
XinHua News	news	-2684.85	-2326.39	-2352.62	-2527.78	-2317.47
ZaoBao News	editorial	-2376.12	-2047.55	-2116.42	-2235.79	-1943.25
Avg. Perplexity		123.83	60.54	68.41	107.57	43.71

Table 2.71: Likelihoods of unseen documents under HM-BiTAMs, in comparison with competing models.

2.7.1.6.2 Application 1: Bilingual Topic Extraction

Monolingual topics: HM-BiTAM facilitates inference of the latent LDA-style representations of topics made by Blei *et al.* (2003), in both English and the foreign language (i.e., Chinese), from a given bilingual corpora. The English topics (represented by the topic-specific word frequencies) can be directly read-off from the HM-BiTAM parameters β . As discussed in Section 2.7.1.3, even though the topic-specific distributions of words for the Chinese corpora are not directly encoded in the HM-BiTAM, we can marginalize over all possible alignments of the parallel data to synthesize them based on the monolingual English topics and the topic-specific lexical mapping from English to Chinese.

Table 2.72 shows five topics, in both English and Chinese, learned via HM-BiTAM. The top-ranked frequent words in each topic exhibit coherent semantic meanings; and there are also consistencies between the word semantics under the same topic indexes across languages. Under HM-BiTAM, the two respective monolingual word distributions for the same topic are statistically coupled due to sharing of the same topic for each sentence-pair in the two languages. Whereas, if one merely apply LDA to the corpora in each language separately, such coupling can not be exploited. This coupling enforces consistency between the topics across languages. However, like general clustering algorithms, topics in HM-BiTAM, are not necessarily to present obvious semantic labels.

For instance, Topic-”sports” is about the sports-meetings for handicapped people and Topic-”energy” is about the resources needs for the quick economic development in China. Secondly, the semantic labels are highly parallel between English and Chinese: the exact parallel nature on topic assignment captured by our HM-BiTAM.

Topic-Specific Lexicon Mapping: Word-to-word translation lexicons (lexical mappings) between languages are invaluable resource for multilingual natural language processing. IBM Models, however, do not exhibit enough power to disambiguate the translation candidates, because they solely rely on the cooccurrence counts from the training data. Once the topic/subject is changed, IBM models can not adjust accordingly for choices for word translations. Contrary to IBM models, our proposed BiTAM models

present a very natural way to handle switching topics and subjects via topic mixtures.

“sports”		“housing”		“takeover”	
English	Chinese	English	Chinese	English	Chinese
teams	人(people)	house	住房(house)	Chongqing	国家(countries)
sports	残疾(handicapped)	construction	房(house)	company	重庆(Chongqing)
disabled	体育(sports)	government	九江(JiuJiang)	takeover	厂(factory)
games	事业(career)	employee	建设(construction)	Shenzhen	天津(Tianjin)
members	水(water)	living	澳门(macao)	Tianjin	政府(government)
people	世界(world)	provinces	元(Yuan)	City	项口(project)
cause	区(region)	Macau	职工(workers)	national	国有(national)
water	新华社(Xinhua)	Anhui	口前(current)	government	深圳(Shenzhen)
national	队员(team memeber)	yuan	国家(national)	project	兼并(take over)
handicapped	记者(reporter)		省(province)	companies	收购(buy)

“stocks”		“energy”	
English	Chinese	English	Chinese
Shenzhen	深圳(shenzhen)	gas	公司(company)
Singapore	深(shen)	company	天然气(gas)
HongKong	新(Singapore)	energy	两(two)
Stock	元(Yuan)	usa	国(countries)
National	股(stock)	Russia	美(U.S.)
Investment	香港(Hongkong)	France	记者(reporters)
Yuan	国有(state-owned)	Chongqing	关系(relations)
options	外资(foreign investment)	resource	俄(Russian)
million	新华社(Xinhua)	China	法(France)
dollar	融资(refinancing)	economy	重庆(Chongqing)

Table 2.72: Monolingual topics of both languages learned from parallel data. It appears that the English topics are highly parallel to the Chinese ones (annotated with English gloss).

Topics	TopCand	“meet” Meaning	Probability	TopCand	“power” Meaning	Probability
Topic-1	运动会	sports meeting	0.5085	电力	electric power	0.5657
Topic-2	满足	to satisfy	0.1602	电厂	electricity factory	0.6560
Topic-3	适应	to adapt	0.9211	涉及	to be relevant	0.9853
Topic-4	调整	to adjust	0.9969	力量	strength	0.4105
Topic-5	会见	to see someone	0.6937	力量	strength	0.9976
Topic-6	-	-	-	-	-	-
Topic-7	满足	to satisfy	0.4676	瓦	Electric watt	0.6137
Topic-8	运动会	sports meeting	0.4877	实力	power	1.0000
Topic-9	-	-	-	输	to generate	0.5046
Topic-10	会见	to see someone	0.5515	力量	strength	1.0000
IBM Model-1	运动会	sports meeting	0.5903	电厂	power plant	0.3143
HMM	运动会	sports meeting	0.7220	力量	strength	0.5149
IBM Model-4	运动会	sports meeting	0.6084	力量	strength	0.5063

Table 2.73: Topic-specific translation lexicons learned by HM-BiTAM. We show the top candidate (TopCand) lexicon mappings of “meet” and “power” under ten topics. (The symbol “-” means nonexistence of significant lexicon mapping under that topic.) Also shown are the semantic meanings of the mapped Chinese words and the mapping probability $p(f|e, k)$.

Table 2.73 shows two examples of topic-specific lexicon mapping learned by HM-BiTAM. Given a topic assignment, a word usually has much less translation candidates and the topic-specific translation lexicons are generally much smaller and sharper.

Different topic-specific lexicons emphasize different aspects of translating the same source words, which cannot be captured by the IBM models or HMM. This effect can be observed from Table 2.73.

Empirical Measures of Lexicon Distributions: We can further investigate the translation lexicons by looking into three measures. First, each English word invokes a simplex and the entropy can be defined for each English word, as in the following equation. The entropy reflected how confident the lexicon $p(f|e, k)$ is on the translations of a given English word e .

$$H(f|e) = \sum_{f \in V_F} p(f|e) \cdot \log p(f|e). \quad (2.80)$$

The second measure we use is the L-2 Norm. This measure is to describe the shape of the lexicon's distribution. It describes the probability mass allocated for the top-ranked translation candidates for a given English word e . If the top-ranked candidates occupy large probability mass, the L-2 Norm will also be large. The large L-2 Norm value indicates the distribution peaks around the top-ranked candidates and the lexicon is considered to be sharp. To compute the L-2 Norm, we have:

$$S(f|e) = \sum_{f \in V_F} p(f|e)^2. \quad (2.81)$$

The third measure we take is the “epsilon-95%”, which is an empirical measure of the “fuzziness” of a lexicon. For a given English word, we count how many of the proposed translation candidates are above the threshold of $p(f|e, k) > 0.05$. This is a rough measure to tell how skewed the lexicon's distribution is.

$p(f e, k)$	topics	entropy	L2-Norm	epsilon-95%
IBM-1	-	-2.20	0.2051	6.10
HMM	-	-1.17	0.3979	3.66
IBM-4	-	-0.92	0.4820	2.99
HM-BiTAM	1	-0.759	0.5740	2.51
	2	-0.754	0.5764	2.48
	3	-0.792	0.5615	2.56
	4	-0.817	0.5491	2.61
	5	-0.834	0.5398	2.65
	6	-0.837	0.5381	2.65
	7	-0.856	0.5299	2.68
	8	-0.879	0.5212	2.74
	9	-0.878	0.5215	2.73
	10	-0.848	0.5338	2.66

Table 2.74 Measuring Translation Lexicons' Size, Distribution shape and Sharpness for IBM Model-1, HMM, IBM Model-4 and HM-BiTAM models.

Table 2.74 shows the average entropy, L2-Norm, epsilon-95% and lexicon's size for baseline IBM models, HMM and HM-BiTAM.

2.7.1.6.3 Application 2: Machine Translation

The *parallelism* of topic assignment between languages modeled by HM-BiTAM, as shown in Section 2.7.1.4 and exemplified in Table 2.72, enables a natural way of improving translation by exploiting semantic consistency and contextual coherency more explicitly and aggressively. Under HM-BiTAM, given a source document D_F , the predictive probability distribution of candidate translations of every source word, $P(e|f, D_F)$, must be computed by mixing multiple topic-specific translation lexicons according to the topic weights $p(z|D_F)$ determined from monolingual context in D_F . That is:

$$P(e|f, D_F) \propto P(f|e, D_F) = \sum_{k=1}^K P(f|e, z = k)P(e|z = k)P(z = k|D_F). \quad (2.82)$$

We used $p(e|f, D_F)$ to score the bilingual phrase-pairs in a state-of-the-art GALE translation system trained with 250M words. We kept all other parameters the same as those used in the baseline. Then decoding of the unseen ten MT04 documents in Table 2.75 was carried out.

Figure 2.56 shows the inferred topic-assignments for the oracle experiments and the documents paired with their hypotheses, respectively. There are obvious differences between the two types of data. First, the document paired with its translations has more small topics inside. The documents paired with human translation usually have less noisy topics. In this sense, the model predictions are closer to human intuitions. The first and the last topics (topic-1 and topic-10) play larger role in the topic-assignment from oracle experiments. Topic 6 is everywhere when we use the translation hypothesis from the baseline system. This is because the errors introduced from the baseline hypothesis are mostly content words and they can change the topic assignment significantly.

Systems	1-gram	2-gram	3-gram	4-gram	BLEU r4n4	NIST	BLEUm4r4c
Baseline SMT	75.63	42.71	25.00	14.30	32.78	7.8749	31.31
Ground Truth	76.10	43.85	26.70	15.73	34.17	7.9333	32.46
Hyp-Doc	76.14	42.74	25.41	14.74	32.95	7.8268	31.46
HM-BiTAM	76.77	42.99	25.42	14.56	33.19	7.8478	31.67

Table 2.75: Decoding MT04 10-documents: Gale systems' output from CMU; Experiments using the topic assignments inferred from oracle document-pairs (Ground Truth); testing monolingual document paired with its English translations (Hyp-Doc); the topical assignments inferred via HM-BiTAM; case-insensitive BLEU score (BLEUn4r4), NIST scores and case-sensitive BLEU score (BLEUm4r4c) are reported.

Table 2.75 shows the state-of-the-art phrase-based SMT Gale-baseline and the HM-BiTAM model, on the NIST MT04 test set. If we know the ground truth of translation to infer the topic weights, the improvement is from 32.78 to 34.17 BLEU points. With topical inference from HM-BiTAM using monolingual source document, improved N-gram precisions in the translation were observed from 1-gram to 4-gram. The largest improved precision is for unigram: from 75.63% to 76.77%. Intuitively, unigrams have potentially more ambiguities for translations than the higher order ngrams, because the later ones encode context information. The overall BLEU score improvement of HM-BiTAM over other systems including the then state-of-the-art is from 32.78 to 33.19.

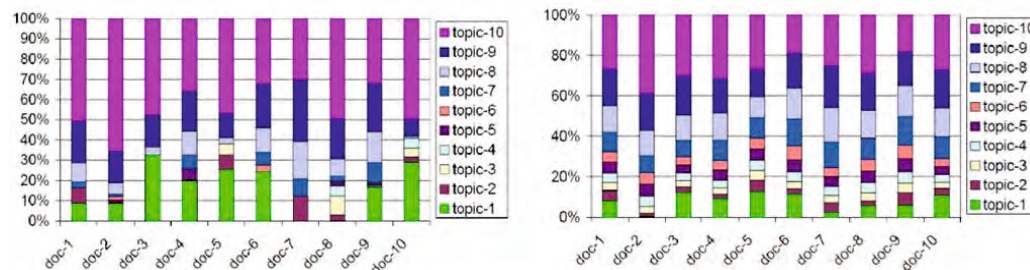


Figure 2.56 Topic-assignment for the 10 testing documents for both oracle scenario (left side: oracle) and LDA scenario using translation hyps(right side: Hyp-Doc).

2.7.1.7 Discussion and Conclusion

In this section, we present a novel framework, **BiTAM**, for exploring hidden bilingual topical aspects given document-level context. In particular, we propose **HM-BiTAM** to integrate HMM’s proximity constraints for improved word-alignment accuracies and translation qualities. Graphical model representations, variational inference and learning for BiTAMs have been developed for efficient model training and applications in translation. Our experimental results show significant improvement of word-alignment accuracies over related methods; we also demonstrate the capability of HM-BiTAM to explore synchronously coherent bilingual topics for parallel data. Encouraging improvement of translation qualities are obtained using admixture of topics inferred via HM-BiTAM. Although the margin is modest, it is noteworthy that the current HM-BiTAM offers more training perplexity reduction across genres. As the parallel data released by LDC become to have richer labels for genre information, we expect further improvements to use the available lightly supervised labels directly in our proposed BiTAM framework.

2.7.2. Active Learning Based Data Selection for Machine Translation

Authors: Arindam Mandal, Dimitra Vergyri, Dilek Hakkani-Tür, Wen Wang, Jing Zheng, Andreas Stolcke, Gokhan Tür and Necip Fazil Ayan

2.7.2.1 Introduction

Statistical machine translation (SMT) models (Brown *et al.* 1990; Koehn *et al.* 2003; Chiang 2007) rely on the availability of parallel data to estimate probabilities of target language sentences given source language sentences. The “goodness” of the estimated probabilities depends on the quality, size and coverage of the parallel corpus. The conventional approach for improving performance of SMT systems is to increase the size of the parallel corpus by adding new training data. Therefore, a significant amount of work has been put into collecting source data and manually translating them into the target language. For instance, within the GALE program (Olive 2005), thousands of

words from different genres have been manually translated each year, with the goal of improving the performance of the SMT systems. However, obtaining large amounts of high quality parallel training data is an expensive process, since human involvement is needed to generate reliable translations.³⁷ Therefore, it is an important issue for the SMT community to be able to select the most beneficial data to be translated, and thus, use the human resources in the most efficient way.

In this study, we propose two new methods to select new source data for manual translation to improve the performance of SMT systems that already achieve reasonable translation performance levels. Our first approach is motivated by active learning techniques (Seung *et al.* 1992) and uses a measure of disagreement between SMT systems to select the most informative training sentences to translate and add to the training corpus. The second method is similar to the n -gram coverage-based selection approach (Eck *et al.* 2005) and uses a perplexity measure to select sentences that seem more different from examples already observed, but robustly avoids selecting rare examples in the domain of interest. Our approaches are intended to choose the *best* source data from a pool of available source sentences and they do not rely on information from the target side. The selected data is not specific to any test set and is intended to generalize to all test sets by choosing data that is different from what is already available. Our experiments show that using either the disagreement-based approach or the perplexity-based approach, translation models trained using only one-fifth of the additional training corpus are able to achieve the same or better performance as those trained using all available additional parallel data, across multiple domains and evaluation sets.

2.7.2.2 Related Work

Prior research on parallel data collection or data selection can be categorized into three groups: automatic data collection, domain/test-set adaptation, and data selection for translation.

Under automatic data collection methods, Resnik *et al.* (2003) identified web pages with similar content that appear in two or more languages, and extracted sentences that are translations of each other from these web pages. Munteanu *et al.* (2005) mined comparable corpora in two languages to identify sentences that are translations of each other. However, these automatic methods are not perfect and generate errorful parallel training data.

The most widely explored area in data selection for SMT is the adaptation of translation models or the language models to a given test set or domain. Hildebrand *et al.* (2005) use information retrieval techniques to select parallel training data that is most similar to a given test set. Lu *et al.* (2007) use an information retrieval system to assign weights to each training sentence pair according to their similarity to the sentences in a given test set, prior to estimating the translation model. Ittycheriah and Roukos (2007) employ a data “sub-sampling” method by choosing the most similar sentences from the

³⁷ Eck *et al.* (2005) report that it costs up to 250,000 USD to translate 1 million words into English from a new language.

training data that have the highest n -gram overlap with a given test set. Adaptation can be performed at the level of the target language model (Zhao *et al.* 2004; Eck *et al.* 2004) by selecting data using information retrieval techniques and using it to adapt the language model.

A final group of prior work focuses on “smart” selection of new source sentences to be translated by humans to improve performance of SMT systems. To our best knowledge, the only work in this area is n -gram coverage-based data selection method proposed by Eck *et al.* (2005), where sentences that have the highest number of previously unobserved n -grams are selected since these would have the greatest impact on coverage. In this study, we follow the third approach and propose two new methods to select data for translation by humans. Our goal is to improve SMT systems by selecting data that is different from what is already available rather than choosing sentences that are similar to a given test set. Thus, our methods are general enough to improve performance on all test sets.

2.7.2.3 Inter-System Disagreement

As mentioned in Section 2.7.2, our approach to selecting parallel training data is motivated by active learning techniques. Active learning has been used in a wide range of applications including spoken language processing (Tür *et al.* 2005), parsing (Hwa 2004) and automatic speech recognition (Hakkani-Tür *et al.* 2002; Kamm and Meyer 2002). One class of active learning approaches introduced by Seung *et al.* (1992) uses a set of distinct learners, each trained with a small number of annotated examples, to label new training examples. The training examples for which the distinct learners agree the least (in labeling) are chosen for careful labeling and used as training data for the learner. This ensures that the learner is provided with the most informative training examples. Such approaches are referred to as *query-by-committee* methods and form the basis of our study. A detailed theoretical analysis of query-by-committee methods for neural networks is reported in Freund *et al.* (1997).

First, we start with a set of N different SMT systems S , which are trained using the same (or similar) initial source language training data corpus \mathcal{D}_i and use them to translate a held-out corpus \mathcal{D}_h . Thus, the SMT systems in S differ only in the algorithms they use in generating hypothesized translations. Next, we estimate the intersystem disagreement described in Algorithm 1.

Finally, we learn a function f (linear regression in this study) that can predict the actual translation performance on new (and untranslated) source language data, \mathcal{D}_u , using measures of intersystem translation disagreement, as described in Algorithm 2. A threshold on the predicted translation performance on sentences in \mathcal{D}_u is used to select candidate sentences for careful translations that can be used as additional parallel training data. We will refer to \mathcal{D}_u as the corpora of interest for the rest of this article.

Algorithm 1: Computing SMT system disagreement

- 1: Start with a set of $S = \{S_1, S_2, \dots, S_N\}$ of N distinct SMT systems. The translation model for each system in S is trained using the same (or similar) set of initial source language sentences \mathcal{D}_i . Given a set of held-out source language sentences \mathcal{D}_h : $\mathcal{D}_h \not\subseteq \mathcal{D}_i$.
- 2: Translate the source sentences in \mathcal{D}_h using each system in S to obtain a set of N translations $\{\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_N\}$
- 3: Compute the translation error \mathcal{X} : $\mathcal{X}_j \forall \mathcal{T}_j \in \mathcal{T}$ using as translation references the translation in each \mathcal{T}_k : $\forall k \in \{1, \dots, N\}; k \neq j$. Each translation error \mathcal{X}_j reflects the degree of disagreement between system S_j and remaining systems in S . These error metrics are referred to as inter-system translation errors.
- 4: Also, evaluate the translation error \mathcal{Y} : $\mathcal{Y}_j \forall \mathcal{T}_j \in \mathcal{T}$ using as translation references the target language side of \mathcal{D}_h (this step is required for Algorithm 2).

Algorithm 2: Data selection based on predicted SMT error

- 1: Start with a set S of N distinct SMT systems (Step 1 of Algorithm 1) and a set of source language sentences \mathcal{D}_u : $\mathcal{D}_u \not\subseteq \mathcal{D}_i$. Given a set of inter-system translation disagreement metrics \mathcal{X} and a set of actual translation error metric \mathcal{Y} (Steps 3 and 4 of Algorithm 1 respectively)
- 2: Learn a function $f: \mathcal{X} \rightarrow \mathcal{Y}$
- 3: Obtain a set of translations for \mathcal{D}_u using each system in S and inter-system translation disagreement metrics \mathcal{U} (same as Steps 3 and 4 of Algorithm 1)
- 4: Use the function f and disagreement metrics \mathcal{U} to predict translation error \mathcal{V}
- 5: Use a threshold on predicted translation error metrics \mathcal{V} to choose the most informative sentences from \mathcal{D}_u

2.7.2.4 Perplexity-Based and Hybrid selection

We also explored an n -gram-based approach for selecting new parallel corpora. This approach required the prior availability of two language models: \mathcal{L}_i , which is trained on source sentences in corpora \mathcal{D}_i and \mathcal{L}_{i+u} , which is trained on source sentences in both \mathcal{D}_i and \mathcal{D}_u . The data selection approach is described in Algorithm 3. The aim of the perplexity ratio metric of Algorithm 3 is to avoid selecting sentences that are rare (outliers) in the corpora of interest \mathcal{D}_u . Sentences that are rare in both \mathcal{D}_i and \mathcal{D}_u , will have high perplexity values (for both \mathcal{L}_i and \mathcal{L}_{i+u}) and a low ratio, while sentences of interest, i.e., the ones with high perplexity (and hence rare) according to \mathcal{L}_i and low perplexity according to \mathcal{L}_{i+u} (not rare in the domain of interest), will have high perplexity ratios. Using a predetermined threshold on perplexity ratio of sentences, candidates for careful translation can be selected from \mathcal{D}_u and subsequently used as additional parallel training data.

Algorithm 3: Data selection based on perplexity

1. Start with a language model \mathcal{L}_i trained on the source language sentences \mathcal{D}_i and a language model \mathcal{L}_{i+u} trained on source sentences in \mathcal{D}_u and \mathcal{D}_i
2. Compute the set of perplexities \mathcal{P}_i of the sentences in \mathcal{D}_u using \mathcal{L}_i and the perplexities \mathcal{P}_{i+u} of the sentences in \mathcal{D}_u using \mathcal{L}_{i+u}
3. Compute the ratio $\alpha = \mathcal{P}_i / \mathcal{P}_{i+u}$ for each sentence in \mathcal{D}_u . Use a threshold on α to choose informative sentences from \mathcal{D}_u

To leverage potential benefits of complementary data sets selected by the intersystem disagreement-based and perplexity methods, we considered an approach that combines the two. In this approach, we select equal amounts of additional training data using each of the previous approaches.

2.7.2.5 Systems and Data Sets

For this study, we focused on Mandarin-to-English translation and used $N = 3$ different SMT systems, each developed independently and achieving competitive translation performance in recent NIST benchmark evaluations. System S_1 (Zheng *et al.* 2008) is based on hierarchical phrase-based translation (Chiang 2007). System S_2 (LanguageWeaver 2005) and System S_3 (Zens 2008) are phrase-based translation systems. For all three systems both translation models and target language models were trained on Mandarin-to-English parallel newswire and web text data sets released by LDC prior to 2005 comprising about 29 million English words.³⁸ These sets correspond to the set \mathcal{D}_i . The corpora of interest for data selection, \mathcal{D}_u consisted of the Mandarin-to-English parallel data released by LDC after 2005, also including both newswire and web text sources and comprised approximately 16 million English words.³⁹ Since the focus of this study was selecting training data for translation models, we kept the target-language model fixed in all experiments. It was trained using the English side of the corresponding parallel data of \mathcal{D}_i corpora, the English Gigaword third edition corpus comprising 3.62 billion words with a vocabulary size 163,941. It used modified Kneser-Ney smoothing (Chen and Goodman 1998) and was trained using the SRILM toolkit (Stolcke 2002).

The proposed data selection methods were evaluated by retraining one of the three systems, S_1 and translating four test sets in use by the GALE program (number of English words in parentheses): Dev 2007 Newswire (18,784), Dev 2007 Web Text (17,556), Eval 2006 Newswire (10,565) and Eval 2006 Web Text (9,953). The log-linear optimization (Och 2003) of the translation models was performed using held-out data comprising 1576

³⁸ The following LDC catalog items were included: DC2002E27, LDC2003E07, LDC2003E14, LDC2004T08, LDC2005E47, LDC2005T06.

³⁹ The following catalog items were included: LDC2006E24, LDC2006E34, LDC2006E85, LDC2006E92, LDC2006G05, LDC2007E08, LDC2007E101, DC2007E103, LDC2007E46, LDC2007E87.

sentences drawn from Dev 2007 Newswire and Dev 2007 Web Text using SRILM tools. All systems are evaluated with the IBM BLEU metric using a single reference.

2.7.2.6 Baseline Systems

	\mathcal{D}_i	$\mathcal{D}_{u+} \mathcal{D}_i$
Dev 2007 Newswire	15.2	16.4
Dev 2007 Web Text	12.6	13.5
Eval 2006 Newswire	16.9	17.2

Table 2.76: Potential lower and upper bounds from including additional parallel data (numbers are Bleu scores)

Our data selection pool \mathcal{D}_u consisted only of parallel corpora released by LDC. Since \mathcal{D}_u has been translated manually, we were able to evaluate the effect of including all of \mathcal{D}_u as training data in addition to \mathcal{D}_i . The results, shown in Table 2.76, can be considered an approximate upper bound on potential performance improvements from data selection.⁴⁰

2.7.2.7 Data Selection Experiments

The next step was to apply the various data selection methods and evaluate their impact on SMT performance. We first describe our experimental configuration for each method.

Inter-System Disagreement

First, we used the three systems in S to translate the sentences in \mathcal{D}_u . The translation performance of each of the three systems is shown in Figure 2.57. As can be seen, the performance of the three systems is quite similar across the different subsets of \mathcal{D}_u , though they disagree, which allows us to learn the function f (Step 2 of Algorithm 2) that predicts translation performance based on inter-system disagreement. Since data selection is performed at the sentence level (and BLEU is not robust for individual sentences), we decided to use Translation Error Rate (TER) (Snover *et al.* 2006) as the target variable to estimate for f . We applied Algorithm 1 by using a rotating subset of \mathcal{D}_u as our held-out set \mathcal{D}_h for estimating f and then applying f on the complement subset, in a jack-knifing fashion. The function f learned was a linear regression over two input variables: the sentence level inter-system TER and the inter-system BLEU score. The linear regression function thus estimated performed better than predicting the mean sentence level TER of \mathcal{D}_u by 10% relative in root mean square error, which was statistically significant at the level of $p = 10^5$.⁴¹ Since we wanted to select certain targeted amounts of data, we did not perform selection by a given threshold (Step 5 of Algorithm 2). Instead, we ranked all

⁴⁰ Assuming that translation performance does not improve by excluding all available data. In practice this might not be true if “bad” or unrepresentative data is included.

⁴¹ This quantifies the margin by which the linear regression is better than predicting chance.

sentences by their predicted TER and chose the sentences with the highest TER up until the desired amount of data was reached.

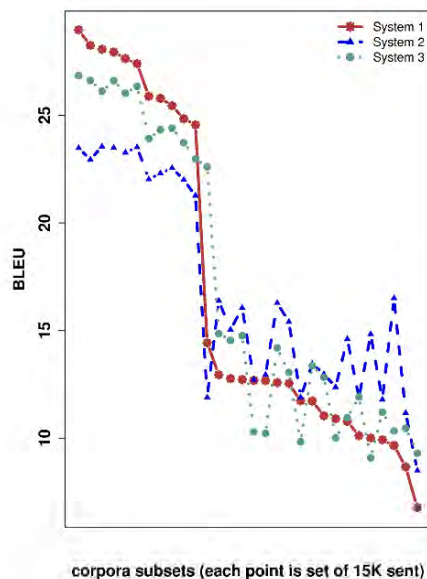


Figure 2.57: Comparison of SMT systems when translating \mathcal{D}_u (Each point corresponds to 15000 sentences)

Perplexity-Based Selection

For the perplexity-ratio-based selection approach, we needed two source-side (Mandarin) language models, \mathcal{L}_u and \mathcal{L}_{i+u} , as described in Section 2.7.2.4. \mathcal{L}_i was trained on 33 million words with a vocabulary size of 60,453 and \mathcal{L}_{i+u} was trained on 55 million words with a vocabulary size of 93,134, both using modified Kneser-Ney smoothing. We followed the steps outlined in Algorithm 3 for selecting parallel training data. Similar to agreement-based selection, all sentences in \mathcal{D}_u were ranked according to their perplexity ratios and the sentences with the highest values were chosen up to the desired training corpus size.

Hybrid Selection

For the combined selection method, equal amounts of data (in terms of number of English words) were selected using the agreement-based and the perplexity-based approaches. Sentences selected by both methods were included only once in training.

Random Selection

This approach involves selecting source language sentences from the corpora of interest \mathcal{D}_u at random as candidates for careful translation and subsequent inclusion in the parallel training corpus. This is equivalent to the chance condition and allows us to determine the effectiveness of our proposed approaches. For the purpose of estimating

robust translation performance metrics, random sampling of parallel training data was repeated several times (four in our experiments) and a translation model was built using each sample. The translation performance metrics were then averaged over all random selection trials to obtain the final performance estimate.

Using the four data selection approaches, we selected varying amounts of additional parallel training data to add to our existing hierarchical phrase-based SMT system.⁴² For each training set, we recomputed phrase alignments, retrained the translation models and reestimated the parameters for log-linear model combination, while keeping the target language model fixed. We compared three sizes of additional data: 4%, 10% and 20% of the available selection pool D_u , corresponding to approximately 640,000, 1.6 million and 3.2 million English words, respectively. The results for the four evaluation sets are shown in Figure 2.58 and Figure 2.59. Each figure includes the result with only initial data set (“Initial”) and with all additional data (“+100%”). Results with increasing intermediate amounts of additional data are shown from left to right. Different shading patterns correspond to different selection methods. The hybrid approach is denoted “agree+ppl”.

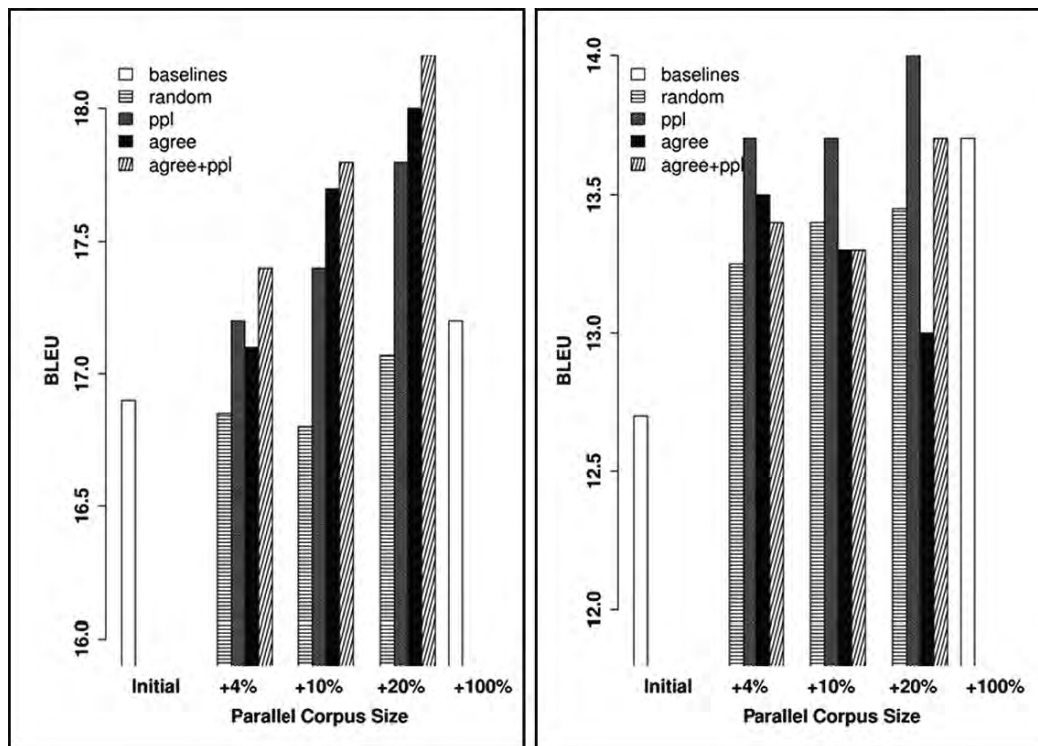


Figure 2.58: BLEU scores for Eval2006 news wire & web text

⁴² Manual translations of selected sentences were available and are used in training.

At 20% additional data, one or more of the proposed selection methods perform better than the putative upper bound (“+100%”), for all four test sets. The perplexity-based approach performs best for Dev 2007 Newswire (by 0.2 BLEU over “+100%”) and for Eval2006 Web Text (by 0.3 BLEU over “+100%”). Hybrid selection performs best for Eval 2006 Newswire (by 1.0 BLEU over “+100%”) and Dev2007 Web Text (by 0.1 BLEU over “+100%”). In addition, it can be observed that with only 10% of additional data, one of the proposed methods still outperforms the all-data system for two of the four evaluation sets. The hybrid method performs best for Eval 2006 Newswire (by 0.6 BLEU over “+100%”) and perplexity-based selection matches the performance of the “+100%” case for Eval 2006 Web Text.

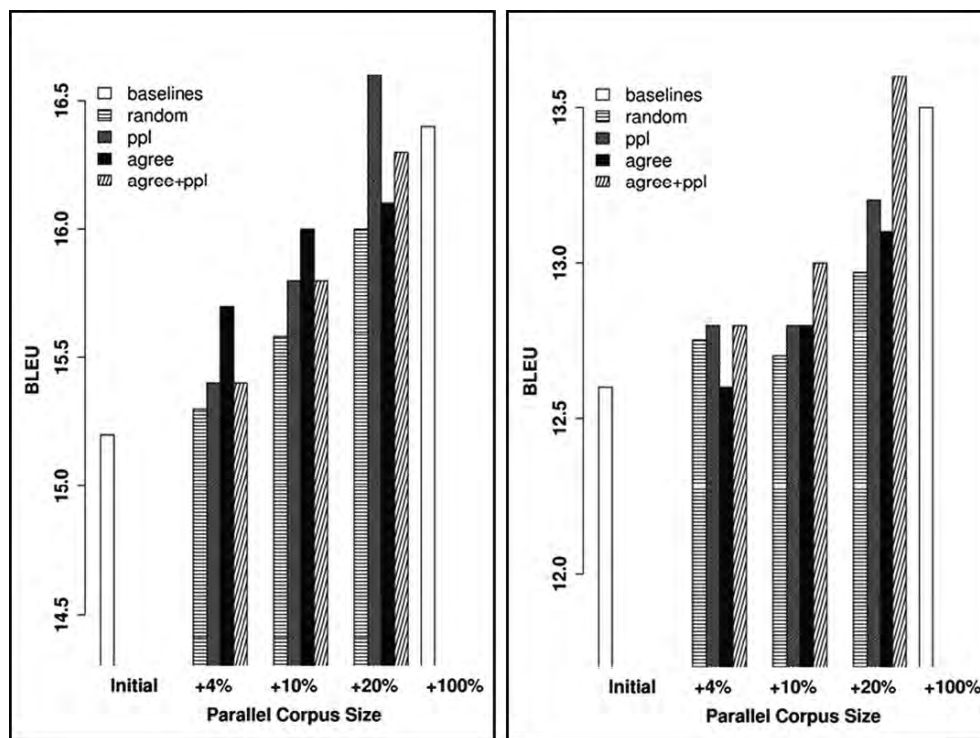


Figure 2.59: BLEU scores for Eval2007 news wire & web text

The pattern in these results, achieving better performance with fewer training samples, agrees with previously reported results using active learning techniques by Schohn *et al.* (2000) and Tür *et al.* (2005). When comparing the different selection methods, a broad generalization that emerges is that the perplexity-based method works better on web text, while the disagreement-based method works better on newswire. As might be expected, the hybrid approach avoids problems in cases where one of the methods alone has relatively poor results; in some cases it gives substantial gains over each of the simpler methods.

2.7.2.8 Conclusions and Future Work

We have proposed three approaches for selecting parallel training data for translation models. The first approach is based on using a measure of inter-system translation disagreement to select data that is most difficult for existing SMT systems. The second approach uses a source-side language model perplexity measure to select sentences that have novel information while avoiding outliers with respect to the domain of interest.⁴³ The third approach combines the first two by selecting equal amounts of data using both methods. In our experiments, these approaches are able to select 20% of available data to achieve Mandarin-to-English translation accuracy that is similar or better than training on the full data set.

The inter-system translation disagreement-based approach suggests some interesting research directions. In particular, we intend to retrain the distinct SMT systems (used for measuring disagreement) with the additional parallel training data (selected using our approach) and using the retrained systems for further data selection. This procedure would be closer to traditional active learning techniques.

2.8 System Combination

2.8.1. Combination of Machine Translation Systems via Hypothesis Selection

Authors: Almut Silja Hildebrand and Stephan Vogel

2.8.1.1 Introduction

In the field of machine translation, systems based on different principles for the generation of automatic translations such as phrase based, hierarchical, syntax based or example based translation have advanced to achieve similar translation quality. The different methods of machine translation lead to a variety of translation hypotheses for each of the source sentences. This diversity can be leveraged by system combination methods to improve translation quality.

Our method combines n -best lists from all input systems and then selects the best hypothesis according to several feature scores on a sentence to sentence basis. Our method is independent of internal translation system scores. Besides the n -best list, no further information from the input systems is needed, which makes it possible to also include non-statistical translation systems in the combination.

2.8.1.2 Features

All features described in this section are calculated based on the translation hypotheses only. We calculate all feature scores for all hypotheses from all systems in a

⁴³ When translating data selected using our approaches, human translators have to perform translation at the sentence-level. This may result in additional effort since the standard practice is to translate entire documents.

consistent manner. In preliminary experiments we added the system scores to our set of features in the combination. This did not improve the combination result and in some cases even hurt the performance, probably because the scores and costs used by the individual systems are generally not comparable.

Using only system independent features enables our method to use the output of any translation system, no matter what method of generation was used there. The system independence of our features also leads to a robustness regarding varying performance of the individual systems on the different test sets used for tuning the feature weights and the unseen test data.

The particular features are:

1. Language Models
2. Statistical Word Lexica
3. Position Dependent N -best List Word Agreement
4. Position independent N -best List N -gram Agreement
5. N -best List N -gram Probability
6. Sentence Length Features
7. System Bias

The first five features are described in Hildebrand and Vogel (2008). We will describe the remaining two here.

Sentence Length Features

To enable the system to avoid selecting too short, as well as, too verbose translations we added a set of three sentence length features.

The length ratio between the source and the target sentence length, the deviation from the overall source-target token count ratio in the bilingual training corpus and the deviation of the sentence length from the average hypothesis length of all hypothesis in the n -best list for this source sentence.

Adding the plain sentence length as feature did not prove useful in preliminary experiments.

System Bias

We also trained global weights to bias the individual systems. We used tuning data from the exact same systems we combine for the unseen data. We use a simple binary feature indicating which system each hypothesis came from and determine the weights in MERT (Och 2003) together with the other feature weights.

2.8.1.3 Experiments

2.8.1.3.1 Evaluation

In this section we report results using BLEU (Papineni *et al.* 2002) and TER (Snover *et al.* 2005) metrics. In the MER training (Och 2003) we optimize for maximum BLEU.

For the Chinese to English task the test sets from the NIST MT evaluations in 2003 and 2006 were used as development and unseen test data. The MT03 test set contains 919

sentences of newswire data with four reference translations. From the 2006 NIST evaluation we used the text translation portion of the NIST part of the test data, which consists of 1099 sentences with four references of newswire and weblog data. The test sets for Arabic-English, GALE-DEV07 (tuning) and NIST-MT08 (unseen) both contain newswire as well as weblog data. For each result reported in the following sections we used MER training to optimize the feature weights on an n -best list for the respective tuning set. These always contained the same set of systems and were combined under the same conditions as for the unseen data.

2.8.1.3.2 Models

In all of the following experiments we used two language models trained from different sources, six features from a pair of statistical word lexica, three features from the position dependent n -best list word agreement and six features each from the n -best list n -gram agreement as well as the n -best list n -gram probability, 23 features in total.

The language models were trained on the data from all sources in the English Gigaword Corpus V3, which contains several newspapers of the years between 1994 to 2006, observing the blackout dates for all NIST test sets. We also included the English side of the bilingual training data, resulting in a total of 2.7 billion running words after tokenization.

From these corpora we trained two language models. A 500 million word 4-gram LM from the bilingual data plus data from the Chinese Xinhua News Agency and an interpolated 5-gram LM from the complete 2.7 Gigaword corpus.

For the latter we trained separate open vocabulary language models for each source and interpolated them using the SRI Language Modeling Toolkit (Stolcke 2002). Held out data for the interpolation weights was comprised of one reference translation each from several NIST-MT test sets. Table 2.77 shows the interpolation weights for the different sources. Apart from the English part of the bilingual data, the newswire data from the Chinese Xinhua News Agency and the Agence France Press have the largest weights. This reflects the makeup of this development test data, which comes in large parts from these sources. Other sources, for example the UN parliamentary speeches or the New York Times, differ significantly in style and vocabulary from the test data and, therefore, get small weights.

Source	Weight	Source	Weight	Source	Weight	Source	Weight
xin	0.30	Bil	0.26	afp	0.21	un	0.07
cna	0.06	Apw	0.05	nyt	0.03	ltw	0.01

Table 2.77: LM interpolation weights per source (source ids as listed in the LDC catalog)

The statistical word lexica were trained respectively on the Chinese-English or Arabic-English bilingual corpora relevant to GALE available through the LDC.⁴⁴ After sentence alignment and data cleaning for Chinese these sources add up to 10.7 million sentences with 260 million running words on the English side and for Arabic we used 7.7

⁴⁴ <http://projects ldc.upenn.edu/gale/data/catalog.html>

million sentences with 217 million running words on the English side. The lexica were trained with the GIZA++ toolkit (Och and Ney 2003).

The research groups who provided the system outputs all had access to the same training data, there is no additional data used in the combination step.

2.8.1.3.3 Chinese-English

Systems

We used the output from six different Chinese-English machine translation systems trained on large data for the GALE evaluation in the beginning of 2008. They are based on phrase based, hierarchical and example based translation principles, trained on data with different Chinese word segmentations, built by three translation research groups, running four MT decoders. The systems A to F are ordered by their performance in BLEU on the unseen Chinese MT06 test set (see Table 2.78).

system	MT03	MT06 BLEU / TER
A	34.68	31.45 / 59.43
B	35.16	31.28 / 57.92
C	34.98	31.25 / 57.55
D	34.70	31.04 / 57.20
E	33.50	30.36 / 59.32
F	28.95	26.00 / 62.43

Table 2.78: Individual Chinese systems sorted by BLEU on unseen data

Feature Impact

For the Chinese-English combination tasks we used following feature groups:

- language models
- word lexicon
- word agreement
- n -gram agreement
- n -gram probability

For comparison to this set of 23 features we ran our setup with the two language models only as a simple baseline. In Och *et al.* (2004) word lexicon features were described as the most useful features for n -best list rescoring. Thus, we added those to the language model probabilities as a second baseline (LM+Lex). The results in Table 2.79 show that a hypothesis selection, which uses these models alone cannot improve over the BLEU score of 31.45 of system A. This probably is the case, because the statistical systems among the input systems are already using this type of information and in fact share the training data which was used to build those models.

To explore the question which feature group contributes the most to the improvement in translation quality and to avoid testing all possible combinations of features, we removed one feature group at a time from the complete set. Table 2.79 shows that although adding the word lexicon features to the language models did not improve the result for the LM+Lex baseline, the overall result still drops slightly from 33.72 BLEU for

all features to 33.61 BLEU for noLex. The combination result decreases insignificantly, but consistently when removing any feature group.

Features	MT03	MT06 BLEU / TER
LM only	35.13	31.17 / 59.34
LM+Lex	36.96	30.97 / 59.41
no LM	39.10	32.83 / 56.23
no Lex	39.62	33.61 / 56.88
no WordAgr	39.59	33.67 / 57.25
no NgrAgr	39.45	33.47 / 56.58
no NgrProb	39.69	33.65 / 57.40
LM+NgrAgr	39.45	33.58 / 57.15
All	39.76	33.72 / 56.79

Table 2.79: Impact of feature groups on the combination result

The biggest drops are caused by removing the language model (-0.89 for no LM) and the n -gram agreement (-0.25 for no NgrAgr) feature groups. Using only those feature groups, which have the biggest impact brings the combination result up to 33.58 BLEU, which is close to the best, but using all features still remains the best choice.

N-Best List Size

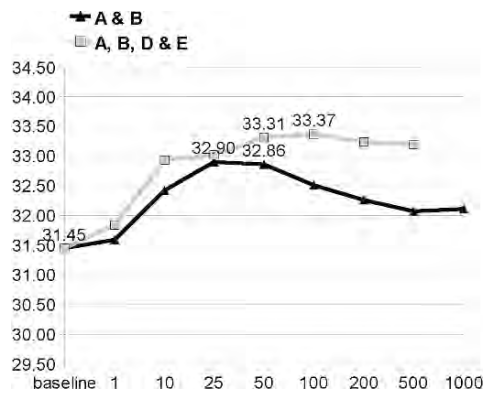


Figure 2.60: Combination results for different n -best sizes for two and four systems for MT06 in BLEU

To find the optimal size for the n -best list combination, we compared the results of using list sizes from 1-best up to 1000-best for each individual system. Hasan *et al.* (2007) investigated the impact of n -best list size on the rescoring performance. They tested n -best list sizes up to 100,000 hypotheses. They found, that using more than 10,000 hypotheses does not help to improve the translation quality and that the difference between using 1000 and 10,000 hypotheses was very small. Based on their results we decided not to go beyond 1000-best.

Because unique 1000-best lists were only available from the systems A, B, D and E, we ran two series of experiments using the top two systems as well as these four systems. In the combination of two systems the n -best list sizes of 25 and 50 hypotheses achieve virtually the same score (See Figure 2.60). The same is true for sizes 50 and 100 in the combination of four systems.

The optimal size of the n -best list seems to be around 50 and also we only had unique 50-best lists available for one of the six systems, we chose the n -best list size of 50 hypotheses for all our following experiments.

The reason why the optimal n -best list size is rather small could be due to the fact that the input lists are ordered by the producing systems. Including hypotheses lower in the list introduces more and more bad hypothesis along with some good candidates. The restriction of the n -best list to the small size of 50 could be interpreted as indirectly using the knowledge of the decoder about the quality of the hypotheses, which is represented in the rank information.

Combination of All Systems

Starting with rescoring the n -best list of system A by itself, we progressively added all systems to the combination. The results in Table 2.80 show, that adding systems one by one improves the result with smaller impact for each additional system.

We achieved the best result of 33.72 BLEU by combining five systems, which is a gain of 2.27 points over the best system. The BLEU score on the tuning set MT03 for this combination was 38.63, which is 3.2 points higher than the best score on MT03 of 35.16 by system B. Adding the weakest system, with more than 5 BLEU points distance to the best system, does not improve the combination in the BLEU metric. However, even though system F also has the highest TER, the combination including all six systems could reduce TER by -0.75 over the best baseline TER of 57.20 by system D (see Table 2.80). TER is consistently reduced with adding more systems, even though all individual systems as well as the combination were optimized to maximize BLEU. Further results for recent GALE systems and additional test data can be found in Section 2.8.2.

System	baseline	combined BLEU/TER
A	31.45	31.76 / 58.95
+ B	31.28	32.86 / 57.90
+ C	31.25	33.32 / 56.87
+ D	31.04	33.51 / 56.77
+ E	30.36	33.72 / 56.79
+ F	26.00	33.63 / 56.45

Table 2.80: Combination results for adding in all systems progressively for MT06 in BLEU/TER

2.8.1.3.4 Arabic-English

Systems

We used the output from seven different large scale Arabic-English machine translation systems trained in October 2008 for the GALE project. The automatic evaluation scores for the systems (see Table 2.81) span 5 BLEU points and about 4 TER points for DEV07 as well as MT08, which were both unseen test sets for the individual

systems. The systems A to G are ordered by their performance in BLEU on the GALE DEV07 test set, which was unseen data for the systems and was used as tuning set for the combination.

system	DEV07	MT08	MT08-NW	MT08-WB
A	50.18 / 43.07	47.47 / 45.21	54.34 / 40.20	38.06 / 51.64
B	49.07 / 45.95	46.33 / 47.59	52.61 / 42.52	37.27 / 54.08
C	47.46 / 45.82	44.50 / 48.09	50.78 / 43.02	36.15 / 54.59
D	47.26 / 46.22	45.07 / 48.17	51.15 / 43.06	36.30 / 54.74
E	45.82 / 44.97	44.42 / 47.54	51.01 / 41.42	35.82 / 55.40
F	45.76 / 46.52	43.35 / 49.35	48.41 / 45.28	35.66 / 54.58
G	45.42 / 47.76	42.49 / 49.65	48.67 / 44.89	33.80 / 55.76

Table 2.81: Individual Arabic-English systems BLEU/TER, sorted by BLEU on unseen data

Combination Result

system	DEV07	MT08	MT08-NW	MT08-WB
combi 7sys	52.43 / 42.39	48.79 / 45.16	55.79 / 39.79	39.47 / 52.05
+ length	52.96 / 42.20	49.15 / 45.20	56.01 / 39.69	39.99 / 52.26
+ system bias	53.01 / 41.85	49.40 / 45.01	56.14 / 39.38	40.39 / 52.22

Table 2.82: Hypothesis Selection results for Arabic-English in BLEU/TER

We combined all seven Arabic-English translation systems and achieved the best result of 49.40 BLEU using all feature groups, which is an improvement of +1.93 BLEU over the best single system. The combination is most successful on the harder web log genre with an improvement of +2.33 BLEU over the best system (see Table 2.82).

The two new feature groups we introduced for the Arabic-English task each improve the combination result insignificantly, but the cumulative improvement of +0.6 BLEU on the whole unseen test set, and especially the +0.92 BLEU on the web log genre is positive. Further results with improvements of up to 3.1 BLEU for recent GALE systems and additional test data can be found in Section 2.8.2.

Hypothesis Length

For the Arabic-English combination task we added the sentence length feature group to the features used for Chinese-English. Table 2.82 shows, that adding these features improves translation quality slightly by +0.36 BLEU, with the slightly bigger contribution of +0.52 BLEU coming from the web log part of the MT08 data.

translation	MT08-NW	MT08-WB
shortest	52.34 / 0.975	40.41 / 0.942
longest	48.41 / 1.000	36.16 / 0.986
combi 7 sys	56.07 / 0.995	40.49 / 0.975
+ length	56.13 / 0.998	40.86 / 0.979

Table 2.83: BLEU Precision/Length penalty for MT08

A look at the BLEU precision and length penalty in Table 2.83 reveals that the improvement in BLEU for the combination using the sentence length feature group is not simply due to a better length penalty.

Simply longer hypotheses usually hurt n -gram precision. The length penalty is in fact only changed by less than half a percent, while the BLEU n -gram precision improves slightly. The sentence length features seem to be able to avoid the selection of simply longer hypothesis just to lower the length penalty.

System Bias

Table 2.82 shows that the system bias improves translation quality slightly by +0.25 BLEU compared to the experiment including the sentence length features. There is a more visible improvement of +0.4 BLEU on the more difficult web log genre. This might be the case because this data is out of domain for the statistical models, therefore, the model scores are not as reliable for this data and the combination is more dependent on the system bias.

Figure 2.61 shows the percentile contributions of hypotheses to the combined translation. The diagrams show clearly how radically the composition of the combined translation changes with the use of the system bias, but surprisingly this does not affect the automatic MT metrics much at all.

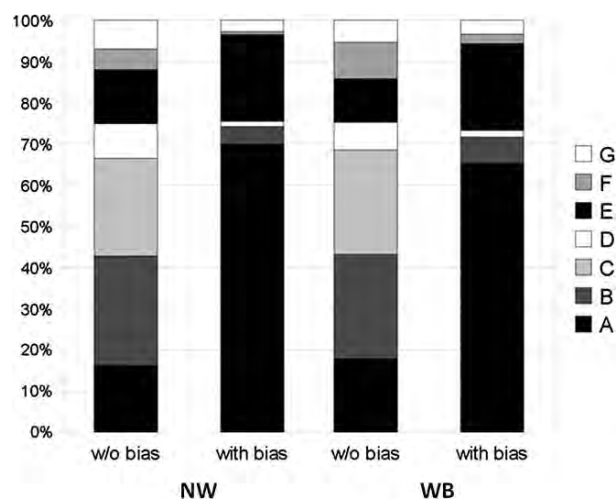


Figure 2.61: Contribution of each system to the combination with and without system bias, MT08-NW and MT08-WB

2.8.1.3.5 Cross Combination

Within the collaboration of the GALE project a reranked n -best list from the hypothesis selection system was made available to a confusion network based system combination (see Section 2.8.2) as additional input system. As unseen test data we used part of the Chinese GALE Phase 3 development set, combined with the Chinese test set of the 2008 NIST MT evaluation, consisting of 1350 segments of text and 1480 speech utterances.

The hypothesis selection output was chosen as skeleton for the CN generation. The choice of the skeleton influences the word order of the combined translation. The results in Table 2.84 show, that the hypothesis selection cannot outperform the CN based approach, especially if you take the TER results into consideration. The CN based system clearly benefits from the hypothesis selection as additional input system, all results, but one, improve for both metrics in the cross combination.

system	NW	WB	BN	BC
Best system	31.37 / 56.01	23.21 / 58.69	30.36 / 56.86	24.10 / 61.00
Hyp. Sel.	34.46 / 55.67	25.50 / 59.99	32.39 / 56.23	26.83 / 62.56
CN-based	34.21 / 53.77	25.36 / 56.56	31.85 / 53.52	27.38 / 58.71
Cross-comb	36.16 / 52.56	26.88 / 57.15	33.61 / 53.15	28.40 / 58.69

Table 2.84: Cross-combination results for Chinese-to-English text and ASR on DEV+MT08 in BLEU/TER.

2.8.1.4 Conclusions

We introduced a system combination method for machine translation systems. In our approach we select hypotheses from a joint n -best list of all input systems, using sentence level features calculated independently from the individual systems internal features. We combine n -best lists from several machine translation systems for the large scale Chinese to English and Arabic to English translation tasks of the GALE project in 2008 and achieve an improvement of +2.3 BLEU in Chinese and +1.9 BLEU in Arabic over the best individual system.

The language model and n -best list n -gram agreement feature groups have the biggest impact on the performance of the system combination. Removing any feature group decreases the BLEU score.

In our experiments the optimal n -best list size lies between 25-best and 100-best. Adding more systems to the combination improves the result progressively as long as the original system performance is not too much below the best system.

The two feature groups we introduced for the Arabic-English task each improve the combination result insignificantly, but the cumulative improvement of +0.6 BLEU on the whole unseen test set and especially the +0.92 BLEU on the web log genre is positive.

Even though the translation quality for the two language pairs is in a very different range, the hypothesis selection technique works for both. The hypothesis selection output proved useful as skeleton for the CN based system combination, improving the result by one BLEU point or more for all genres.

Our method can easily be extended to use more feature scores, for example additional models like phrase tables, class based language models and distortion models.

2.8.2. Confusion Network Decoding for MT System Combination

Authors: Antti-Veikko I. Rosti, Evgeny Matusov, Jason Smith, Necip Fazil Ayan, Jason Eisner, Damianos Karakos, Sanjeev Khudanpur, Gregor Leusch, Zhifei Li, Spyros Matsoukas, Hermann Ney, Richard Schwartz, Bing Zhang, and Jing Zheng

2.8.2.1 Introduction

Confusion network decoding has been very successful in combining speech-to-text (STT) outputs (Fiscus 1997; Evermann and Woodland, 2000; Mangu *et al.*, 2000) from diverse systems using different modeling assumptions. Several modeling paradigms have been introduced in machine translation (MT) including rule-based, phrase-based, hierarchical, syntax-based and even cascades of rule-based and statistical MT systems. Building confusion networks from MT system outputs is more challenging compared to STT system outputs since the translations may have very different word orders and varying lexical choices without affecting the meaning of the sentence, whereas, the words and the word order of speech transcriptions are strictly defined by the utterance.

A confusion network is a linear graph where all paths travel via all nodes. There may be one or more word arcs between two consecutive nodes. These arcs may be viewed as alternative choices of words in a hypothesis. Thus, a confusion network may encode an exponential number of hypotheses. A word arc may also contain a NULL word which represents an empty word or a deletion. Fiscus (1997) aligns the STT outputs incrementally to form a confusion network. The vote count of each word arc is increased by one for each matching word in the alignment. The path with the highest total number of votes through the lattice defines the consensus output. Simple edit distance is sufficient in building confusion networks from STT system outputs since the outputs should follow a strict word order defined by the actual utterance. The most common STT quality metric, word error rate, only considers exact matches as correct. The order in which the STT system outputs are aligned does not significantly influence the resulting network. In machine translation, there may be several correct outputs with different word orders, as well as, different words or phrases with identical meaning. Two problems not relevant to combining STT system outputs arise: how to align outputs with different word orders and how to choose the word order of the final output. Many alignment algorithms for building confusion networks from MT system outputs have been proposed including edit distance based multiple string alignment (Bangalore *et al.* 2001), hidden Markov model based alignments (Matusov *et al.* 2006; He *et al.* 2008), inversion transduction grammar (ITG) (Wu 1997) based alignments (Karakos *et al.* 2008) and translation edit rate (TER) (Snover *et al.* 2006) based alignments (Sim *et al.* 2007; Rosti *et al.* 2007a; Rosti *et al.* 2007b; Rosti *et al.* 2008; Ayan *et al.* 2008; Rosti *et al.* 2009). Alignment algorithms based on TER approximations using both TERCOM and ITGs and symmetric word alignment from a hidden Markov Model (HMM) are detailed in this section. One MT hypothesis must be chosen to be the “skeleton” against which all other hypotheses are aligned. The final word order and the quality of the decoding output depend on the skeleton selection.

The MT system combination approaches based on confusion network decoding consist of the following steps:

1. Collect N -best list outputs from MT systems and optionally perform some preprocessing (lower case, re-tokenize, etc.);
2. Choose one or more skeleton translations for each segment;
3. Align all other hypotheses against the skeleton/skeletons;
4. Build confusion network/networks from the alignments for each segment;
5. Decode confusion networks using arc features and sentence-level scores (LM score, word insertion score, etc.);
6. Optimize feature weights on a development set;
7. Optional post-processing (true casing, detokenization, etc.).

In incremental hypothesis alignment algorithms, steps 3 and 4 are performed in a single process.

The following three sections describe the confusion network decoding approaches used by all three teams participating in the most recent DARPA GALE evaluations. Three TER based alignment algorithms and various skeleton selection methods developed by the AGILE team are presented in Section 2.8.2.2. Section 2.8.2.3 presents a comparison of two alignment algorithms developed by the NIGHTINGALE team. Finally, an alternative alignment algorithm for TER scoring and hypothesis alignment based on ITGs developed by the Rosetta team is described in Section 2.8.2.4.

2.8.2.2 AGILE Team Approach

Three TER based MT hypothesis alignment algorithms and how to build confusion networks from these are described in this section. The method used to assign scores for paths given system and language model weights and how to choose the skeleton hypothesis are presented. Weight tuning algorithm based on Powell's method is briefly reviewed. Experiments combining outputs from up to nine systems on Arabic newswire, web and audio sets are presented.

2.8.2.2.1 Pair-wise TER Alignment

The TER alignment may be used in building confusion networks from MT hypotheses (Sim *et al.* 2007). The availability of TERCOM has made it easy to create a high performance confusion network decoding baseline (Rosti *et al.* 2007a).⁴⁵ Since TERCOM was developed for automatic evaluation of MT outputs, it can only align two strings, the reference and the hypothesis. Assuming one output has been chosen to be the confusion network skeleton, all hypotheses may be aligned independently using the skeleton as the reference translation. These pair-wise alignments may be consolidated to form a confusion network.

⁴⁵ <http://www.cs.umd.edu/~snover/tercom/> (current version 0.7.25)

Given three strings: “twelve cars”, “twelve big blue cars” and “dozen blue cars”, six pair-wise alignments may be generated. The alignment between the first and second hypotheses is:

NULL	twelve	NULL	cars
NULL	twelve	big blue	cars

The initial NULLs were inserted to help forming the confusion network with the alignment below. This alignment has two insertion errors from the string “big blue”. The alignment between the first and third hypotheses is:

NULL	twelve	NULL	cars
dozen	blue	NULL	cars

This alignment has one insertion from the word “dozen” and one substitution from the word “blue”. The NULLs before the word “cars” were inserted to help forming the confusion network with the previous alignment.

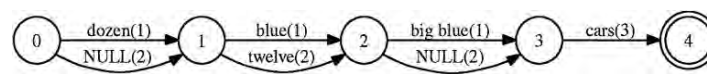


Figure 2.62: Confusion network using pair-wise TER alignment and “twelve cars” as the skeleton.

A confusion network generated from these two alignments is shown in Figure 2.62. The number of votes for each word is denoted by the number in parentheses. The path with the highest vote count is “twelve cars”, but there are some undesirable paths due to alignment errors. Since the pair-wise alignments are independent, there is no guarantee that the word “blue”, which is not present in the skeleton, will align correctly from the other two hypotheses. Multiple insertions “big blue” are aligned with a single NULL arc from the other hypotheses. Aligning multiple insertions from several alignments is non-trivial requiring an additional round of alignments and skeleton selection. Finally, the TER algorithm does not know that words “twelve” and “dozen” are semantically identical.

Using the second string as the skeleton yields the following two alignments:

twelve	big	blue	cars
twelve	NULL	NULL	cars

with two deletion errors and

twelve	big	blue	cars
NULL	dozen	blue	cars

with one deletion and one substitution error. A confusion network generated from these two alignments is shown in Figure 2.63.

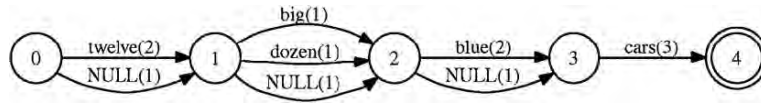


Figure 2.63: Confusion network using pair-wise TER alignment and “twelve big blue cars” as the skeleton.

There are three paths with equal number of votes resulting in hypotheses: “twelve big blue cars”, “twelve dozen blue cars” and “twelve blue cars”. Obviously, the second path is undesirable, although as likely as the other two.

Using the third string as the skeleton yields the following two alignments:

NULL	dozen	blue	cars
NULL	NULL	twelve	cars

with one deletion and one substitution and

NULL	dozen	blue	cars
twelve	big	blue	cars

with one insertion and one substitution error. A confusion network generated from these two alignments is shown in Figure 2.64. Again, there are three paths with equal number of votes resulting in hypotheses: “big blue cars”, “dozen blue cars” and “blue cars”. However, there are some very bad paths such as “twelve dozen twelve cars”, as a result of alignment errors.

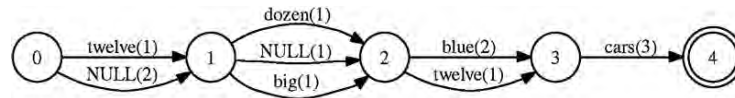


Figure 2.64: Confusion network using pair-wise TER alignment and “dozen blue cars” as the skeleton.

2.8.2.2.2 Incremental TER Alignment

Confusion networks from MT outputs may be built incrementally. Bangalore *et al.* (2001) used a multiple string alignment (MSA) software package to build the networks. MSA (Durbin *et al.* 1998) is based on edit distance alignment and does not perform reordering. An incremental TER alignment allowing shifts was proposed by Rosti *et al.* (2008). First, a trivial confusion network is generated from the skeleton by creating a word arc for each skeleton token. Each hypothesis is aligned against the current network at a time. All words, including the NULL word, on the arcs connecting two consecutive nodes are available for matching with a zero edit cost. A new node and two new arcs are created for each insertion. One of these arcs represents the NULL with votes from all hypotheses aligned so far and the other arc represents the inserted word from the new

hypothesis. A deletion will either create a new NULL arc or increase the vote of an existing NULL arc if one already exists at that position. A substitution will always create a new word arc. Insertions and deletions have a unit edit cost and a substitution cost of $1 + \epsilon$ ($\epsilon = 0.0001$ in the experiments) was found to improve the quality of the decoding outputs.

Unlike in the pair-wise TER alignment, the order in which the hypotheses are added influences the resulting confusion network. As in MSA, the alignment order may be determined by the edit distance from the current confusion network. The hypothesis closest to the network is aligned first. The TER between the current confusion network and all remaining unaligned hypotheses is computed. This requires a total of $0.5(N_s^2 - N_s)$ alignments per segment to be computed for building a network from N_s outputs. Given outputs $\mathcal{E} = \{E_1, \dots, E_{N_s}\}$, an algorithm to build a set of N_s confusion networks $\mathcal{C} = \{C_1, \dots, C_{N_s}\}$ may be written as:

```

for n = 1 to  $N_s$  do
   $C_n \leftarrow \text{Init}(E_n)$  {initialize confusion network from the skeleton}
   $\mathcal{E}' \leftarrow \mathcal{E} - E_n$  {set of unaligned hypotheses}
  while  $\mathcal{E}' \neq \emptyset$  do
     $E_m \leftarrow \text{argmin}_{E \in \mathcal{E}'} \text{Dist}(E, C_n)$  {compute edit distances}
     $C_n \leftarrow \text{Align}(E_m, C_n)$  {align closest hypothesis}
     $\mathcal{E}' \leftarrow \mathcal{E}' - E_m$ 
  end while
end for

```

Using the same strings from the examples in Section 2.8.2.2.1 and the first string as the skeleton, yields the following alignment:

twelve	NULL	NULL	cars
twelve	big	blue	cars
dozen	NULL	blue	cars

where the third hypothesis is an ϵ farther from the initial network compared to the second hypothesis. The confusion network is shown in Figure 2.65. The path with the highest number of votes yields “twelve big blue cars” and there are no undesirable paths. Using the second string as the skeleton yields the following alignment:

twelve	big	blue	cars
twelve	NULL	NULL	cars
dozen	NULL	blue	cars

where again the third hypothesis is an ϵ farther from the initial network compared to the first hypothesis. The confusion network is identical to the one in Figure 2.65.

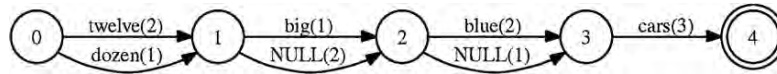


Figure 2.65: Optimal confusion network from incremental TER alignment using “twelve cars” and “twelve big blue cars” as skeletons. The same network results from the incremental alignment with flexible matching described in Section 2.8.2.2.3 using any hypothesis as the skeleton.

Using the third string as the skeleton yields the following alignment:

NULL	dozen	blue	cars
NULL	NULL	twelve	cars
twelve	big	blue	cars

where both hypotheses are an equal distance from the initial network and were aligned in the arbitrary order they were listed. The word “twelve” in the first hypothesis is aligned with the word “blue” since the algorithm does not know that it is synonymous with “dozen”. The string “blue cars” from the second hypothesis is matched exactly with the network, “twelve” appears as an insertion and “big” as a substitution. A confusion network generated from this alignment is shown in Figure 2.66. There are three hypotheses with equal number of votes: “dozen blue cars”, “blue cars” and “big blue cars”. Some undesirable, lower cost paths also exist; for example, “twelve dozen twelve cars”.

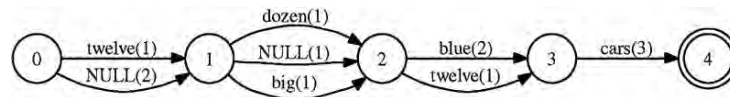


Figure 2.66: Confusion network using incremental TER alignment and “dozen blue cars” as the skeleton.

The incremental TER alignment may be used to extract an approximate TER oracle translation from a confusion network. This can be achieved by aligning a reference translation with the confusion network and finding a path with the lowest edit cost with respect to the aligned reference through the network. This may be repeated for multiple reference translations and the lowest cost path among all reference translations will be the oracle translation.

2.8.2.2.3 Incremental Alignment with Flexible Matching

Ayan *et al.* (2008) and He *et al.* (2008) have proposed improved pair-wise alignment algorithms by trying to match semantically equivalent words. Snover *et al.* (2009) similarly extended the TER algorithm to produce an evaluation metric called TER-plus (TERp). Rosti *et al.* (2009) modified incremental TER alignment to allow matching synonyms and words with identical stems with a lower cost. Semantically equivalent words may be collected from the individual system outputs using WordNet (Fellbaum

1998). During the alignment, substitutions between equivalent words are assigned an edit cost of 0.2. In the above example, there are two equivalent words “twelve” and “dozen” which do not always align when using the incremental TER alignment. Using the first string as the skeleton, the incremental alignment with flexible matching yields the following alignment:

twelve	NULL	NULL	cars
dozen	NULL	blue	cars
twelve	big	blue	cars

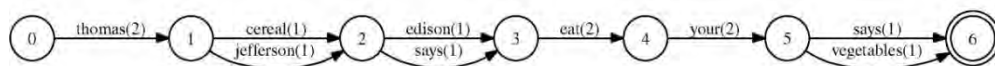
where the distance of the third hypothesis is 1.2 and second hypothesis is 2 from the initial network. Therefore, the third hypothesis is aligned first creating a new node and two arcs with a NULL from the skeleton and “blue” from the third hypothesis. Aligning the second hypothesis also creates a new node and two arcs with a NULL from the other two hypotheses and “big” from the second hypothesis. Using the second string as the skeleton, yields the following alignment:

twelve	big	blue	cars
dozen	NULL	blue	cars
twelve	NULL	NULL	cars

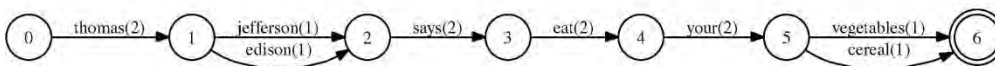
where the word “dozen” aligns correctly with the skeleton word “twelve”. Using the third string as the skeleton, yields the following alignment:

dozen	NULL	blue	cars
twelve	NULL	NULL	cars
twelve	big	blue	cars

where again the words “dozen” and “twelve” are correctly aligned. Each hypothesis used as the skeleton yields the same confusion network shown in Figure 2.65 with the same total edit costs.



(a) Alignment using the standard TER shift heuristics.



(b) Alignment using the modified shift heuristics.

Figure 2.67: Confusion networks using different shift heuristics in alignment.

The heuristic shift constraints of TER are not reasonable when using the flexible matching since there may be a block of tokens in the reference with some matching words and some synonyms. Snover *et al.* (2009) relaxed the shift constraints in computing the TERp score by considering synonyms and words with identical stems as exact matches while using the TER shift heuristics, but using tuned edit costs for these when computing the edit distance. The shift heuristic may be modified to consider any block shifts that do not increase the edit cost.⁴⁶ This increases the complexity of the algorithm considerably, but has some desirable influence on hypothesis alignment. Karakos *et al.* (2008) showed an example where the TER yields a poor alignment. The hypotheses were “thomas jefferson says eat your vegetables” and “eat your cereal thomas edison says”. The TER shifts two blocks “eat your” and “thomas” and the final alignment has three substitution, a total of five edits. The modified algorithm shifts only one block “eat your cereal” resulting in only two substitution errors, a total of three edits. Confusion networks resulting from these alignments are shown in Figure 2.67 a) and b), respectively. Since TERp requires exact, synonym, stem matches, or phrase matches, it would not find the optimal shift in this example unless there are some unlikely phrase pairs in the TERp phrase table. The modified shift heuristics yield clearly a better confusion network for this artificial example. However, it is hard to quantify whether it yields better system combination performance (Karakos *et al.* 2008).

2.8.2.2.4 Word Posteriors and LM Features

As seen in the previous sections, the confusion networks may often contain several paths with an equal number of votes. Also, some MT systems may consistently produce better translations in which case their vote should count more. This may be achieved by introducing system weights. The diversity of the confusion networks may also be increased by aligning N -best hypotheses from the MT systems. In addition, a language model may be able to break any remaining ties by giving a higher score to more fluent hypotheses.

J =3	S =2	E =3	SC =(0.76,0.00,0,0.66,0.10,0.46,0.24,0,0)	W=head
J =4	S =2	E =3	SC =(0.24,0.05,0,0.00,0.00,0.00,0.00,0,0)	W=NULL
J =5	S =2	E =3	SC =(0.00,0.95,1,0.15,0.90,0.41,0.70,1,1)	W=president
J =6	S =2	E =3	SC =(0.00,0.00,0,0.19,0.00,0.00,0.06,0,0)	W=chief
J =7	S =2	E =3	SC =(0.00,0.00,0,0.00,0.00,0.13,0.00,0,0)	W=chairman

Figure 2.68: Example lattice snippet showing five word arcs with word posteriors between nodes 2 and 3. Here J is the arc index, S is the start node index, E is the end node index, SC is a vector of nine system specific word posteriors and W is the word label.

If 1-best outputs are used in the incremental alignment algorithm, a total of $0.5(N_s^2 - N_s)$ alignments are performed, where N_s is the number of MT systems. The N -best

⁴⁶ Snover et al. (2009) investigated relaxing the shift heuristics in the same way. This resulted in lower TERp scores, similarly to using ITGs to estimate TER in section 2.2.1.4 but the correlations with human judgment were worse.

hypotheses from a MT system are usually very similar, so it may be reasonable to align all hypotheses from one system in the rank order and determine the order in which the N -best lists from different systems are aligned by the distance of the 1-best from the current confusion network. This will add at most $N_s(N - 1)$ alignments.

An ad hoc weighting of $1/(1 + n)$ where n is the hypothesis rank is used to give a lower vote for words from lower ranking hypotheses. These votes are collected for each system separately, so that each word arc has N_s scores. After all hypotheses are aligned, the N_s vote counts are scaled to sum to one across all word arcs between two consecutive nodes. An example lattice snippet is shown in Figure 2.68. There are five word arcs between nodes 2 and 3 with the words *head*, *NULL*, *president*, *chief* and *chairman*. Nine system specific scores are shown in parentheses. Each score may be viewed as a posterior probability of the system generating the word in that position. For example, the probability that system 1 generated the word *head* between nodes 2 and 3 is 0.76, the probability for *NULL* is 0.24 and zero for other words.

Given system weights w_n , which sum to one and system specific word posteriors s_{nj} for each arc j , the weighted word posteriors are defined as:

$$s_j = \sum_{n=1}^{N_s} w_n s_{nj} \quad (2.83)$$

The hypothesis score is defined as the sum of the log-posteriors along the path which is linearly interpolated with a log-LM score and a non-NULL word count:

$$S(E|F) = \sum_{j \in \mathcal{J}(E)} \log s_j + \gamma S_{LM}(E) + \delta N_w(E) \quad (2.84)$$

where $\mathcal{J}(E)$ is the sequence of arcs generating the hypothesis E for the source sentence F , $S_{LM}(E)$ is the Log-LM score and $N_w(E)$ is the number of non-NULL words. The set of parameters $\theta = \{w_1, \dots, w_{N_s}, \gamma, \delta\}$ can be tuned so as to optimize an evaluation metric on a development set as described later in Section 2.8.2.2.6. Decoding with an n -gram language model requires expanding the lattice to distinguish paths with unique n -gram contexts before assigning the LM scores to each arc.

2.8.2.2.5 Skeleton Selection

As seen in Sections 2.8.2.2.1 and 2.8.2.2.2, the decision of which skeleton to use may affect the quality of the resulting confusion network. If the alignment algorithm yields better alignments, the decision is less important. The optimal alignment was found for all skeletons in the example in Section 2.8.2.2.3. However, the incremental alignment with flexible matching cannot always find the optimal alignment for longer hypotheses with different word orders.

Sim *et al.* (2007) proposed using the minimum Bayes risk (MBR) criterion with TER as the loss function and a uniform posterior distribution:⁴⁷

⁴⁷ MBR (Kumar and Byrne, 2004) is measured as the minimum expected loss $L(E, E')$ over the posterior probability distribution $P(E|F)$ of all possible translations: $\hat{E} = \operatorname{argmin}_{E'} \sum_E P(E|F)L(E, E')$. Since the posterior probability distributions $P(E|F)$ for translations from different MT systems are usually not comparable, a uniform distribution is assumed in skeleton selection.

$$\hat{E} = \operatorname{argmin}_{E_i} \sum_{j=1}^{N_s} \operatorname{TER}(E_j, E_i) \quad (2.84a)$$

where the MBR hypothesis \hat{E} has the minimum expected TER with respect to other hypotheses E_j . Using the example in Section 2.8.2.2.1, the total number of edits for each hypothesis as the skeleton is four but the second hypothesis is longer than the other two and yields a lower MBR score. Therefore, the hypothesis “twelve big blue cars” would be chosen to be the skeleton. The MBR skeleton is chosen before tuning the system weights, so the weights do not influence the decision.

Matusov *et al.* (2006) proposed selecting the skeleton as late as possible by building N_s confusion networks with the 1-best outputs as skeletons and connecting these networks into a joint lattice with a common start and end node. The system weight was used as a prior probability for each sub-network in the lattice. Rosti *et al.* (2007a) extended this by estimating priors from the MBR scores by treating the expected TER score percentages as negative log-probabilities which were scaled so as to define a prior distribution. These priors were also multiplied by the system weights, so that a skeleton with a high MBR score, but a low system weight would less likely be selected.

2.8.2.2.6 Weight Tuning

The optimization of the system and LM feature weights may be carried out using N -best lists as in the work of Ostendorf *et al.* (1991). Standard tools may be used to generate N -best hypotheses including word confidence scores and language model scores from the lattice representing multiple confusion networks. The N -best list is reordered based on the sentence-level scores $S(E_{m,k}|F_m)$ from Equation 2.84 for the m^{th} source sentence F_m and the corresponding k^{th} hypothesis $E_{m,k}$. The current 1-best hypothesis \hat{E}_j given a set of weights $\theta = \{w_1, \dots, w_{N_s}, \gamma, \delta\}$ may be represented as follows:

$$\hat{E}_m(F_m|\theta) = \operatorname{argmax}_{E_{m,k}} S(E_{m,k}|F_m) \quad (2.85)$$

The objective is to optimize the 1-best score on a development set given a set of reference translations. For example, estimating weights to minimize TER between a set of 1-best hypotheses $\hat{\mathcal{E}}$ and reference translations \mathcal{E}_r can be written as:

$$\hat{\theta} = \operatorname{argmin}_{\theta} \sum_{j=1}^{N_s} \operatorname{TER}(\hat{\mathcal{E}}, \mathcal{E}_r) \quad (2.86)$$

This objective function is not differentiable, so gradient-based optimization methods may not be used. In this work, Powell’s method, as described in the work of Press *et al.* (2007), is used. The algorithm explores better weights iteratively starting from a set of initial weights. First, each dimension is optimized using a line minimization algorithm utilizing bracketing and parabolic interpolation. Then, the direction of largest decrease is replaced with a new direction based on the old directions weighted by the change in the objective function value. This procedure is iterated until the change in the objective function value becomes small. To improve the chances of finding a global optimum, 19 random perturbations of the initial weights are used in parallel optimization runs. Since the N -best list represents only a small fraction of all hypotheses in the confusion network, the optimized weights from one set of optimization runs may be used to generate a new

N -best list from the lattice for the next optimization iteration. The N -best lists are merged between these iterations. Similarly, weights to optimize other evaluation metrics such as BLEU (Papineni *et al.* 2002) or METEOR (Lavie and Agarwal 2007) may be tuned.

Och (2003) proposed a more efficient algorithm for log-linear models where an exact minimum may be computed in the line minimization. Since the system weights are not log-linear interpolation weights in Equation 2.84, the exact line minimization may not be used.

2.8.2.2.7 Experimental Evaluation

The Agile team ran system combination experiments on Arabic Newswire development sets to contrast the effects of different design decisions. Outputs from nine systems covering rule-based, phrase-based, hierarchical, syntax-based and a cascade of a rule-based and a hierarchical MT systems were used. All outputs had up to 10-best hypotheses, were lower cased and used identical tokenization. Unpruned bigram and 5-gram language models, trained on about 8 billion words of English text, were used in confusion network decoding and N -best list rescoring.⁴⁸ A development set of 2118 segments was used in tuning the system combination weights to maximize the BLEU score and a set of 1031 segments was used as a test set.⁴⁹ The development sets consisted of documents from the newswire portion of NIST MT04, MT05, MT06 and MT08 evaluation sets, the GALE P1 and P2 evaluation sets and GALE P2 and P3 development sets. Four reference translations were available for both sets. All individual statistical MT systems were tuned to maximize the BLEU score on a separate tuning set.

System	dev		test	
	TER	BLEU	TER	BLEU
A	48.73	38.38	49.52	38.29
B	41.41	51.08	42.14	49.41
C	40.57	53.03	40.97	51.37
D	40.10	53.64	40.91	51.58
E	38.10	54.79	39.61	52.39
F	37.76	54.96	38.70	52.56
G	38.25	55.21	39.47	52.89
H	37.59	55.69	38.92	53.91
I	37.46	56.25	38.53	54.33
Pair-wise	34.87	59.80	36.08	57.33
Incremental	34.10	61.06	35.00	58.85
Flexible	34.13	60.90	35.25	58.66

Table 2.85: Case insensitive TER and BLEU scores on the Arabic Newswire development and test sets.

Table 2.85 shows the scores for the individual system outputs (systems A through I sorted according to increasing BLEU score), as well as, the system combination outputs using the three different alignment algorithms on the tuning and test sets. The

⁴⁸ Using a trigram LM in decoding did not yield improvements after 5-gram rescoring.

⁴⁹ The brevity penalty was calculated using the original formula in the work of Papineni *et al.* (2002), rather than the definition implemented in the NIST mteval-v11b.pl script.

combination using pair-wise TER alignment yields significant gains over the best individual system scores (system I) on both sets. The incremental TER alignment yields significant gains over the pair-wise TER alignment and similar scores with the incremental alignment with flexible matching. Table 2.86 shows the average number of nodes and arcs per segment in the confusion networks and the scores for the oracle hypotheses extracted from the confusion networks built using the incremental alignment algorithms on the test set. The oracle scores are better when no flexible matching is used. This may be explained by the fact that the flexible matching yields more compact networks which contain fewer paths. The average number of tokens in the test set outputs was 37.83 and the maximum length of a hypothesis that can be generated from a confusion network is equal to the number of nodes minus one.

System	#nodes	#arcs	TER	BLEU
Incremental	57.45	122.87	14.71	78.30
Flexible	53.31	111.48	15.46	76.97

Table 2.86: Average number of nodes (#nodes) and arcs (#arcs) per segment and case insensitive TER and BLEU scores for the oracle outputs on the Arabic Newswire test set.

The pair-wise TER alignments were obtained using the Java TER software and the incremental confusion network building software was written in C++. The average time to build the confusion networks was about 3.58 seconds per segment when using the pair-wise TER alignment, 10.19 seconds per segment when using the incremental TER alignment and 101.72 seconds per segment when using the incremental alignment with flexible matching. The N -best lists had on average 8.92 hypotheses per segment. Using the N -best lists slowed down the alignment by about a factor of two compared to using only 1-best outputs. The gain from using the N -best lists was about 0.4 BLEU points on the development set and the scores on the test set were practically identical.

System	dev		test	
	TER	BLEU	TER	BLEU
Pair-wise	35.77	58.20	36.48	56.06
Incremental	34.58	58.79	35.45	56.89
Flexible	34.67	58.87	35.68	56.62

Table 2.87: Case insensitive TER and BLEU scores on the Arabic Newswire development and test sets without system weights or language model; i.e., voting.

Table 2.87 shows the scores obtained with equal system weights without using a language model. This corresponds to simple voting. Comparing these scores to those in Table 2.85 shows that the system weights, as well as, the bigram decoding and 5-gram rescoring yield about 1.3 to 2 BLEU point gains. The BLEU score differences between the pair-wise and incremental alignment algorithms are greater after tuning. The confusion networks built using the incremental alignment algorithms probably have paths with more fluent hypotheses and the language model helps to find these.

Table 2.88 shows the influence of the skeleton choice on the quality of the decoding output. If only a single system is always chosen as the skeleton, the best scores are

achieved by using system I. The MBR skeleton yields a small, but consistent gain in both TER and BLEU scores on both development and test sets. The following three results are obtained using all 1-best outputs as skeletons and connecting the resulting confusion networks with common start and end nodes. The initial NULL arcs connecting the start node to the confusion networks have unit system specific scores in the first case, no weights. In this case, the system weights do not influence the skeleton selection. In the second case weights, the system specific scores are set to zeros apart from that for the system used as the skeleton. This corresponds to using the system weights as the prior distribution. In the third case MBR priors, a MBR based prior estimate is placed as the system specific score. The TER and BLEU scores on both sets improve consistently from MBR to MBR prior in the order listed. The oracle skeleton was selected by choosing the systems with the lowest TER when used as the skeleton for each segment. This gives an estimate of the upper bound for any skeleton selection method.

Skeleton	dev		test	
	TER	BLEU	TER	BLEU
system I	34.47	60.13	35.64	57.86
MBR	34.30	60.47	35.51	58.25
no weights	34.46	60.58	35.40	58.31
weights	34.25	60.62	35.32	58.48
MBR priors	34.13	60.90	35.25	58.66
oracle	31.41	62.59	32.51	60.19

Table 2.88: Case insensitive TER and BLEU scores for different skeleton choices.

System	WB		BN		BC	
	TER	BLEU	TER	BLEU	TER	BLEU
Best	48.38	42.00	46.62	33.26	50.18	30.12
Pair-wise	46.95	43.81	44.63	34.20	48.04	31.18
Incremental	45.84	46.06	43.92	35.10	47.47	31.76
Flexible	45.14	46.15	44.14	34.89	47.19	32.15

Table 2.89: Case insensitive TER and BLEU scores on the Arabic web, BN and BC test sets.

Finally, results on test sets for web, broadcast news (BN) and broadcast conversations (BC) are shown in Table 2.89. The web test set had mostly four reference translations whereas the audio sets (BN and BC) had only a single reference translation. The BN and BC system outputs were generated by running MT on the 1-best output of the AGILE P3 Arabic STT system. Separate development sets were used to tune the individual MT systems and the system combination weights for newswire, web and audio. Tuning separate system combination weights for BN and BC did not yield any improvements. The system combination weights for the web and audio sets were tuned to minimize $0.5TER + 0.5(1 - BLEU)$ since it yielded significantly lower TER scores without a significant impact on the BLEU scores.

2.8.2.3 NIGHTINGALE Team Approach

The NIGHTINGALE team developed two confusion network system combination approaches that differ in the methods for alignment of individual system hypotheses. The two alignment strategies will be described and directly compared in this section.

The first method is a statistical alignment framework first described by Matusov *et al.* (2006) in which the alignment between words in the target (English) language is learned statistically using a Hidden Markov alignment model (HMM). The context of a whole corpus of parallel system translations rather than a single sentence is considered in this iterative, unsupervised procedure, yielding a more reliable alignment. The second alignment algorithm of Ayan *et al.* (2008) is a two-pass approach: In the first pass, all hypotheses are aligned to the skeleton independently using a TER-based alignment with word synonym matching and a confusion network is built. Next an intermediate reference sentence is created from the confusion network generated in the first pass via majority voting for each position in the network. The second pass uses this intermediate reference as the skeleton translation to generate the final confusion network.

Although the system combination approaches of these two methods differ in the word alignment algorithms used, they share the same CN structure, so that the alignment component can be easily interchanged. Therefore, the differences of the two approaches in scoring the CNs and their dependency on a particular alignment type can be investigated experimentally. We will show that both alignment approaches perform similarly well, resulting in high-quality system combination translations on GALE Chinese-to-English text and speech translation tasks (2008 evaluation).

2.8.2.3.1 RWTH Hypothesis Alignment Approach

The alignment approach of RWTH is a statistical one. It takes advantage of multiple translations for a whole corpus to compute a consensus translation for each sentence in this corpus. It also takes advantage of the fact that the sentences to be aligned are in the same language.

The network is created by the pair-wise method (Section 2.8.2.2.1). The word alignment is *trained* in analogy to the alignment training procedure in statistical MT. We use the IBM Model 1 (Brown *et al.* 1993) and the Hidden Markov Model (Vogel *et al.* 1996) to estimate the alignment model. The alignment training corpus is created from a test corpus of effectively $M \cdot (M - 1) \cdot N$ sentences translated by the involved MT engines.⁵⁰ The single word based lexicon probabilities $p(e|e')$ are initialized from normalized lexicon counts collected over the sentence pairs (E_m, E_n) on this corpus. Since all of the hypotheses are in the same language, we count co-occurring identical words; i.e. if $e_{m,j}$ is the same word as $e_{n,i}$ for some i and j . In addition, we add a fraction of a count for words with identical prefixes.

The model parameters are trained iteratively using the GIZA++ toolkit (Och and Ney 2003). The training is performed in the directions $E_m \rightarrow E_n$ and $E_n \rightarrow E_m$. The updated lexicon tables from the two directions are interpolated after each iteration. The final alignments are determined using a cost matrix C for each sentence pair (E_m, E_n) . The elements of this matrix are the local costs $C(j, i)$ of aligning a word $e_{m,j}$ from E_m to a word $e_{n,i}$ from E_n . Following Matusov *et al.* (2004), we compute these local costs by

⁵⁰ A test corpus can be used directly because the alignment training is unsupervised and only automatically produced translations are considered.

interpolating the negated logarithms of the state occupation probabilities from the “source-to-target” and “target-to-source” training of the HMM model. Two different alignments are computed using the cost matrix C . The first alignment \tilde{a} is a function of words in E_m . It is used to reorder the words in E_m so that the resulting alignment of the reordered E_m with the skeleton E_n becomes monotonic. Then, this alignment is modified to alignment \bar{a} that includes only *one-to-one* connections. In case of many-to-one connections in \tilde{a} of words in E_m to a single word from E_n , we only keep the connection with the lowest alignment costs. The use of the one-to-one alignment \bar{a} implies that some words in the secondary translation will not have a correspondence in the skeleton translation and vice versa. We consider these words to have a null alignment with the empty word ϵ . In the corresponding confusion network, the empty word will be transformed to an epsilon arc. Given the $M - 1$ monotonic one-to-one alignments between E_n as the skeleton translation and $E_m, m = 1, \dots, M; m \neq n$, we construct a confusion network following the approach of Bangalore *et al.* (2001) with an extension that improves the alignment of the words which are insertions with respect to the skeleton hypothesis (Matusov *et al.* 2008).

2.8.2.3.2 SRI Hypothesis Alignment Approach

Another alignment strategy pursued within the Nightingale team by SRI is based on TER alignment extended by matching of synonyms, as described in Section 2.8.2.2.3. In natural language, it is possible to represent the same meaning using synonyms in possibly different positions. For example, in the following sentences, “at the same time” and “in the meantime”, “waiting for” and “expect” and “set” and “established” correspond to each other, respectively:

Skeleton: at the same time expect Israel to abide by the deadlines set by.

Hypothesis: in the meantime, we are waiting for Israel to abide by the established deadlines.

To incorporate matching of word synonyms into the alignment and for better alignment of similar words at different word positions, we employ the following steps:

The first step is to use WordNet to extract synonyms of each word that appear in each hypothesis regardless of their POS tag in the given translation.⁵¹ We should note that WordNet contains only open-class words; i.e., nouns, verbs, adjectives and adverbs. There are no entries for determiners, prepositions, pronouns, conjunctions and particles. For better matching of these additional POS tags, we manually created a different equivalence class for each POS tag that is not included in the WordNet so that words with the same POS tag can be considered synonymous.

⁵¹ Our goal is to add as many synonyms as possible to increase the chances of a word aligning to one of its possible synonyms rather than to any other word. Therefore, we do not distinguish between the synonyms of the same word according to their confidence value or their POS tag.

After extracting the synonyms of each word in the given translations, the next step is to augment each reference word with its synonyms. To avoid over generation of synonyms, we make the assumption that words w_i and w_j are synonyms of each other only if w_i appears in the synonym list of w_j and w_j appears in the synonym list of w_i . In our running example, the augmented (extended) skeleton according to the second hypothesis is as follows:

Extended skeleton: at the [same time \ meantime] [expect \ waiting] Israel to abide by the deadlines [set \ established] by.

The final step is to modify the TER script to favor matching of a word to its synonyms rather than to any other word (see Section 2.8.2.2.3).

Two-Pass Alignment Strategy

When building a confusion network, the usual strategy is first to align each hypothesis to the skeleton separately and reorder them so that the word ordering in the given hypothesis matches the word ordering in the skeleton translation. Next a confusion network is built between all these reordered hypotheses. This approach, however, is problematic when the hypotheses include additional words that do not appear in the skeleton translation. In such cases, two hypotheses other than the skeleton may not align perfectly since the alignments of two different hypotheses are done independently.

To overcome this issue, we employ a two-pass alignment strategy. In the first pass, we align all hypotheses to the skeleton independently and build a confusion network. Next an intermediate reference sentence is created from the confusion network generated in the first pass. To create this intermediate reference, we find the best position for each word that appears in the confusion network using majority voting. The second pass uses this intermediate reference as the skeleton translation to generate the final confusion network.

When we create the intermediate reference, the number of positions for a given word is bounded by the maximum number of occurrences of the same word in any hypothesis. It is possible that two different words are mapped to the same position in the intermediate reference. If this is the case, these words are treated as synonyms when building the second confusion network and the intermediate reference looks like the extended reference above.

2.8.2.3.3 Confusion Network Scoring

In the approaches of RWTH and SRI, given a single skeleton hypothesis, a confusion network is created using one of the two alignment strategies described above. In the RWTH approach, the system combination lattice is a union of several CNs, which were built by considering each of the system translations as the skeleton. The single-best path is determined after summing the probabilities of identical paths which originate from different CNs. This is done through determinization of the lattice. The sum over the

identical hypotheses in the decision criterion can help to reach consensus not only on the word level, but on the level of sentence structure: the word order preferred by the weighted majority of the systems most probably will be used in the system combination translation.

As described above, SRI generates a single confusion network based on a two-pass alignment strategy. Beyond the system prior weights and LM scores, SRI uses two additional features to control the number of epsilon arcs and number of words during the confusion network scoring. The number of words feature is also included in the RWTH system.

The system prior weights and the scaling factors of additional statistical models such as the language model need to be tuned to produce good consensus translations. At RWTH, these parameters are optimized using the publicly available CONDOR optimization toolkit (Vanden Berghen and Bersini 2005). For the GALE evaluation, we selected a linear combination of BLEU and TER as optimization criterion, $\hat{\Theta} := \operatorname{argmax}_{\Theta} (2 \cdot \text{BLEU} - \text{TER})$. In each iteration of the optimization, we extract a new hypothesis directly from the system combination lattice. System combination at SRI was optimized using an in-house implementation of minimum-error-rate training (MERT) (Och 2003) to maximize BLEU score.

2.8.2.3.4 Experimental Evaluation

We present the results of system combination experiments on the Chinese-to-English GALE data which was used by the Nightingale team to prepare for the December 2008 evaluation. The results in the tables below are reported in terms of the well-established automatic MT evaluation measures BLEU and TER; the evaluation was case insensitive, but considered punctuation marks. We used the closest reference length for computation of brevity penalty.

Development and Test Sets

For the experiments, we used all of the available GALE-related data with multiple (4) translations. It includes (parts of) the GALE 2006 and 2007 evaluation sets, the official GALE 2007 development set, as well as the GALE part of the NIST MT 2008 evaluation data. The data was provided by LDC under the catalog numbers LDC2008E08, LDC2008E09, LDC2008E19. To define the development and the blind test sets from this data, we firstly separated all the documents into 4 genres: newswire text (NW), webtext (WT), broadcast news (BN) and broadcast conversations (BC). Then, in each genre the documents were divided in two more or less equally-sized parts with the same distribution of sources. In the following, we refer to these two parts (for all genres) as the NIGHTINGALE test sets A and B.

We combined up to 8 system outputs in our experiments as described below. The systems which produced those outputs followed phrase-based, hierarchical, and syntax-based translation paradigms. One of the systems was a serial combination of a rule-based and a phrase-based MT system. Table 2.90 gives an overview of the number of sentences in the test sets A and B that had to be translated by these systems, as well as, the number of words they have produced. The test set A was used for individual system tuning,

whereas, the test set B was used for tuning the weights of the features in the system combination algorithms. Thus, the test set A can be considered as “blind” test set for judging the performance of the system combination approaches presented here.

	NW	WT	BN	BC
test set A				
Sentences	485	533	529	1134
Running words (K)	15.8	13.7	14.7	16.7
Distinct words (K)	5.5	5.2	4.6	3.7
test set B				
Sentences	480	490	483	937
Running words (K)	15.9	13.1	13.1	14.2
Distinct words (K)	5.5	5.0	4.5	3.4

Table 2.90: Corpus statistics for the hypotheses of the 8 individual systems used in system combination on the NIGHTINGALE test sets. The number of running words is averaged over the 8 systems.

Combination of Variants of a System

Table 2.91 presents BLEU and TER scores for 3 input systems (hierarchical phrase-based system (HPBT) and its two syntax-based extensions, denoted by SAMT and S2D) and their combination using SRI’s system combination approach. All three input systems were trained on the same training data but with different word segmentations and the n -best lists were reranked using different subsets of seven additional LMs. The input systems were optimized on test set A newswire and webtext (text) using minimum error rate training (MERT) to maximize BLEU score. System combination was also optimized on the same test set on 2000-best lists generated from the confusion network decoding, using MERT to maximize BLEU score. As inputs to the system combination, we used 10-best hypotheses from each of the reranked n -best lists. For language model scores during system combination, we used the same 4-gram LM that we used during decoding of the individual systems.

The results indicate that the system combination yields better translations even when the input systems are very similar to each other in terms of their performance: on newswire and webtext data, system combination yields an absolute improvement of up to 1.3 BLEU point and up to 1.3 TER point over the best individual system. These results validate the results reported in an earlier paper using three variants of SRI’s hierarchical system on different test sets (Ayan *et al.* 2008).

System	test set A		test set B		test set A		test set B	
	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER
	Newswire				Webtext			
HPBT	34.2	57.6	33.7	58.2	26.2	63.4	27.2	62.5
S2D	34.5	59.0	34.3	58.7	26.9	63.7	26.6	64.2
SAMT	34.3	59.5	33.0	59.4	26.9	63.8	27.4	62.5
SysComb	35.6	56.9	34.5	56.9	28.2	63.6	28.7	62.9

Table 2.91: scores (in %) for three SRI systems and their combination. System combination was optimized on test set A and tested on test set B.

General vs. Adapted LM Rescoring

By Matusov *et al.* (2008), it was proposed to rescore the system combination CNs with a trigram LM trained on the outputs of the systems involved in system combination. The LM training data in this case were the system hypotheses for the same test corpus for which the consensus translations were to be produced. Using this “adapted” LM for lattice rescoring, thus, gives bonus to n -grams from the original system hypotheses, in most cases from the original phrases. Presumably, many of these phrases have a correct word order, since they are extracted from the training data. We compared the performance of the SRI’s system combination approach using the “adapted” trigram LM or a 4-gram LM trained on large amounts of English text. For this experiment, we used single best outputs from all eight systems and tested the impact of two LMs on four different genres. The system optimization was performed on the text portion of test set B for newswire and webtext and audio portion of test set B for broadcast conversations and news.

Table 2.92 presents the results on the test sets A for each genre. Our results validate the previous finding about the usefulness of “adapted” LMs, yielding an absolute improvement of up to 0.6 BLEU point and up to 1.0 TER point over using the general higher-order LMs. We explain these improvements in part by the fact that the number of n -grams in a system combination CN can be very large and may include those n -grams which have high general LM probability, but are not in any way related to the source sentence. With the absence of any other model that links words and phrases in the CN to the source sentence, the influence of the general LM may be overestimated. The “adapted” LM does not seem to have this problem, since it only gives high probabilities to a limited number of “good” n -grams preselected by the individual systems based on phrase-to-phrase translation scores and general LM scores.

Genre	general LM		adapted LM	
	BLEU	TER	BLEU	TER
NW	38.1	55.1	38.3	54.5
WT	29.2	60.1	29.8	59.2
BN	31.4	60.0	31.9	59.0
BC	28.8	62.5	29.3	61.3

Table 2.92: Comparison of SRI’s system combination approach with eight input systems and 2 different LMs on test set A (the scores are in %). System combination was optimized on test set B.

Combination of Structurally Different Systems

In the following experiments, we compared the alignment approaches of RWTH and SRI using both the RWTH and SRI CN scoring procedures. In these experiments, we combined single-best outputs of eight systems and the “adapted” LMs were used. The parameter tuning on test set B was genre specific for the RWTH system and was performed separately for text and audio in the SRI system.

Table 2.93 compares the automatic MT evaluation measures BLEU and TER on the “blind” test set A. Both system combination approaches significantly improve translation quality with respect to the best individual system being combined; in some cases, the improvements exceed three points in BLEU and four points in TER.

Alignment	CN scoring	BLEU	TER	BLEU	TER	BLEU	TER	BLEU	TER
		Newswire		Webtext		Broadcast News		Broadcast Conversations	
best single system		35.3	58.5	27.2	62.6	29.7	63.6	28.0	63.3
RWTH	RWTH	38.7	54.2	30.3	59.0	32.5	59.2	29.9	61.8
SRI	RWTH	37.0	55.7	29.7	60.0	32.4	59.5	29.8	61.3
RWTH	SRI	37.6	54.1	29.1	59.0	31.9	59.3	29.1	61.2
SRI	SRI	38.3	54.5	29.8	59.2	31.9	59.0	29.3	61.3

Table 2.93: Comparison of word alignment and CN scoring approaches on test set A (the scores are in %). “Best single system” refers to the individual system with the highest BLEU score among eight systems. System combination was optimized on test set B.

On the audio data, SRI and RWTH alignments perform similarly well, but the RWTH CN scoring algorithm has slightly better BLEU scores than the SRI scoring algorithm. For Newswire and Webtext data, the results are inconclusive: the RWTH system combination approach performs best using the RWTH alignment algorithm, the SRI approach exhibits better scores with the SRI alignment algorithm. Also, there are some discrepancies between BLEU and TER: on Newswire, the RWTH approach is better than the SRI approach in BLEU, but worse in TER. A subjective analysis of the differences between SRI and RWTH approach on the Newswire data has shown that the system combination of SRI is more often able to make the right lexical choice, especially of content words. This is most probably due to the explicit matching of synonyms performed in the construction of confusion networks. In contrast, the RWTH approach often produces better sentence structures and has fewer reordering and missing word errors than the SRI approach.

best single system	... why not have the money to spend, an important reason.
system combination	... there is one important reason why did not dare to spend money.
reference translation	... there is an important reason why people have money but don't dare spend it.
best single system	For the reader, a good story is a strong life again.
system combination	For the reader, a good novel is a powerful reappearance of life.
reference translation	As for readers, a good novel is a powerful reproduction of life.
best single system	They kneel on a chair beside the edge, to pray together.
system combination	They knelt down at the edge of a chair beside him, praying together.
reference translation	They then kneeled down on the edge of a chair beside them and prayed together.

Table 2.94: Examples of improved translation quality after system combination (test sets A and B).

Table 2.94 shows some examples of improved translation quality due to system combination in comparison with the translations of the individual system that had overall best translations on this corpus. The examples were drawn from test sets A and B.

2.8.2.4 Rosetta Team Approach

The Rosetta Team developed a system combination approach based on alignments computed using inversion transduction grammars (ITGs). A pair-wise alignment strategy, similar to the one described in Section 2.8.2.2.1, appeared in the work of Karakos *et al.*

(2008) and was later improved in three ways: (i) the pair-wise alignment strategy was replaced by an incremental alignment strategy, similar to the one described in Section 2.8.2.2.2, but with some crucial differences; (ii) the original ITG-based algorithm was sped up significantly with an A* heuristic that uses a finite-state reordering method for computing a lower bound to the ITG alignment cost, and (iii) the same algorithm was extended to allow ITG alignments between confusion networks, instead of strings.

2.8.2.4.1 Approximate Methods for Computing TER

Since computation of TER is an NP-complete problem, TERCOM uses some heuristics to constrain the space of permutations and, thus, compute *an approximation to* TER in polynomial time. The block shifts which are allowed in TERCOM have to adhere to the following constraints: (i) A block cannot be moved if it has an exact match in its current position, and (ii) for a block to be moved, it should have an *exact* match in its new position. These constraints sometimes lead to counterintuitive sequences of edits, as shown in Section 2.8.2.2.3.

We have modified the TERCOM code to relax the above constraints in the following way: the requirement of a *perfect match* between the words in a block and the words it gets aligned to was replaced by a user-specified bound on the (normalized) *number of edits* in the new position. This roughly means that a block which does not have a perfect match in its original position and does not differ in its new (candidate) position by more than $x\%$ of its length, is allowed to move to that position. These constraints revert to the original TERCOM constraints when $x = 0$. As is described later, this modification was done in order to allow alignments between confusion networks and the use of a more flexible cost function than the binary (match/no-match) cost of the original TERCOM.

Inversion Transduction Grammars

The inversion transduction grammar (ITG) formalism (Wu 1997) allows one to view the problem of alignment as a problem of synchronous parsing. Specifically, ITGs can be used to find the optimal edit sequence under the restriction that block moves must be properly nested, like parentheses. That is, if an edit sequence swaps adjacent substrings A and B of the original string, then any other block move that affects A or B must stay completely within A or B, respectively. An edit sequence with this restriction corresponds to a synchronous parse tree under a simple ITG that has one non-terminal and whose terminal symbols allow insertion, deletion and substitution.

The minimum-cost ITG tree can be found by dynamic programming. This leads to *invWER* (Leusch *et al.* 2003), which is defined as the minimum number of edits (insertions, deletions, substitutions and block shifts allowed by the ITG) needed to convert one string to another. As described in Section 2.8.2.4.2, the minimum-*invWER* alignments are used for generating confusion networks.

Using TERCOM and *invWER* to Approximate TER

In this section, we compare TERCOM and *invWER* in terms of how well they approximate TER. Specifically, the two competing alignment procedures were used to estimate the TER between reference (human) translations and machine translation system outputs submitted to NIST for the GALE evaluation in June 2007. The references are the

post edited translations for each system (i.e., these are “HTER” approximations). As can be seen from Table 2.95, for all systems and all language and genre conditions, invWER gives a better approximation to the true TER than TERCOM.⁵² In fact, out of over 16,000 total segments in all languages/genres, TERCOM is lower in less than 1% of the segments, while invWER is lower in over 10%. This is a clear indication that ITGs can explore the space of string permutations more effectively than TERCOM.

Lang. / Genre	System 1		System 2		System 3	
	TERCOM	invWER	TERCOM	invWER	TERCOM	invWER
Arabic BC	18.4%	18.0%	24.6%	24.1%	20.6%	20.4%
Arabic BN	17.8%	17.5%	23.2%	22.8%	19.0%	18.7%
Arabic NW	12.1%	12.0%	14.8%	14.6%	15.4%	15.2%
Arabic WB	22.0%	21.7%	23.6%	23.2%	25.6%	25.3%
Chinese BC	30.7%	29.9%	32.0%	30.9%	33.6%	32.7%
Chinese BN	22.9%	22.4%	25.5%	24.8%	29.5%	28.8%
Chinese NW	20.8%	20.5%	24.3%	23.6%	25.9%	25.2%
Chinese WB	26.2%	25.9%	27.2%	26.5%	30.3%	29.9%

Table 2.95: Comparison of average per-document TERCOM with invWER on the EVAL07 GALE Broadcast Conversation (“BC”), Broadcast News (“BN”), Newswire (“NW”) and Weblogs (“WB”) data sets.

2.8.2.4.2 Incremental System Combination

This section describes the incremental alignment method used in our experiments. We have been inspired by the work of Rosti *et al.* (2008), but we have modified their incremental algorithm to improve speed (without loss in performance).

1. The n -best list of each system is aligned together incrementally into a confusion network using regular edit distance alignments, which are roughly 50 times faster than the TERCOM alignments described later. Thus, several per-system confusion networks are created.⁵³
2. One system (usually the one with the best performance) is chosen as the skeleton. Its confusion network bins are used as anchors for aligning all other confusion networks together.
3. One-by-one, the per-system confusion networks are aligned with the partially generated confusion network. We have experimented mainly with TERCOM and ITG-based alignments during this step.
4. The final probability of each arc is set to the relative frequency of its label in the bin.
5. The final confusion network is then rescored with a 5-gram language model; this results in a reassignment of arc costs according to a loglinear combination of the costs (as computed above) and the probability assigned by the language model. The weights in the linear combination are determined through a development set.

⁵² Both tercomTER and invWER are upper bounds to true TER, so lower is better.

⁵³ The n -best list of a single system will usually only contain local reorderings; we have found that using monotone alignments in this step does not degrade performance.

To extend pair-wise string alignment algorithms to confusion networks, we treat the confusion network bins as “words” and use the probability of mismatch between bins as the word substitution cost. More precisely, the formula for computing the substitution cost between two bins $b_1; b_2$ is⁵⁴

$$\frac{1}{|b_1||b_2|} \sum_{w \in b_1, v \in b_2} n_1(w) n_2(v) \mathbf{1}(w \neq v) \quad (2.87)$$

where $n_j(x)$ is the number of times word x appears in bin j (x can also be the empty word, epsilon). The cost of inserting/deleting a bin is similarly computed as the substitution cost between the bin and an empty bin. When two bins are aligned together to form a new merged bin, the counts of their words get added together. If a bin gets inserted into, or deleted from, the confusion network, a NULL arc is also added in the appropriate place.

Combinations Using TERCOM

To handle confusion network bins, we performed the following modifications to TERCOM:

1. It can take as input a pair-wise cost matrix, which determines the cost of substituting one confusion network bin with another. This allows the use of Equation 2.87 when aligning confusion networks.
2. The modification mentioned in Section 2.8.2.4.1 was extended to confusion network bins and the threshold x was applied to the normalized sum of bin substitution costs, as computed by 2.8.2.4.1. We used a value of $x = 50\%$ in our experiments as a compromise between performance and tractability.

Combinations with Inversion Transduction Grammars

The work on ITG-based incremental alignment is a follow-up on the work of Karakos *et al.* (2008), where ITGs were used for aligning together MT outputs based on the pair-wise alignment procedure of Rosti *et al.* (2007a). Originally, an 11-rule Dyna program (Eisner *et al.* 2005) was used to compute ITG alignments, which we replaced by an optimized C++ code that significantly reduced the execution time. This implementation was sped up 10-fold by an A* search heuristic, whose details appear below. When comparing this new implementation with the modified TERCOM code described above, ITG alignments are roughly five times faster. However, for computing TER on the dataset described in Section 2.8.2.4.1, the ITG code takes three times longer than the unmodified version of TERCOM.

In order to speed up the $O(n^6)$ synchronous parse under the inversion transduction grammar used for alignments, we use best-first chart parsing (Charniak *et al.* 1998).

⁵⁴ This formula can be easily modified to use alternate word substitution costs in place of the exact matching given by $\mathbf{1}(w \neq v)$. We have experimented with the surface similarity of the work of He *et al.* (2008) and character edit distance, but did not see any improvements.

Rather than filling in the chart in a fixed order, we create an agenda of edges and process them in an order based on some measure of “goodness”. To ensure that the first parse we find is optimal, the “goodness” measure must be an optimistic estimate of the total cost of any parse containing that edge. This approach is known as A* parsing (Klein and Manning 2003).

Let c_1 and c_2 be the confusion networks that we are synchronously parsing with the ITG grammar, where $|c_1| = n_1$ and $|c_2| = n_2$ ($|c|$ refers to the number of bins in c). Let e be an edge in the ITG chart which spans the intervals (i_1, j_1) in c_1 and (i_2, j_2) in c_2 . The estimate of the total cost is divided into an inside estimate (the cost of aligning the intervals (i_1, j_1) to (i_2, j_2)) and an outside estimate (the cost of aligning $(0, i_1) \cdot (j_1, n_1)$ to $(0, i_2) \cdot (j_2, n_2)$). The inside cost is already known in bottom-up parsing, so we focus on the outside estimate. Let c'_1 be the confusion network created by concatenating $(0, i_1)$ and (j_1, n_1) , and similarly, let c'_2 be the concatenation $(0, i_2)$ and (j_2, n_2) . The outside estimate can be reformulated as a lower bound of the cost of parsing c'_1 and c'_2 . Also, recall that invWER is an upper bound to TER, so any lower bound of TER will also be a lower bound of invWER. We now describe a finite state reordering scheme which acts as a lower bound to TER.

As mentioned earlier, the TER between two strings is defined as the minimum number of word insertions, deletions, substitutions and block moves required to transform one string into the other. If block moves were not allowed, computation would be simple and in fact equivalent to edit distance. To allow the possible reorderings introduced by block moves, we add “jump” arcs to the edit distance machine which allow free movement in the input string (shown in Figure 2.69). Shapira and Storer (2002) show that a sequence of n block moves splits a string into at most $3n + 1$ contiguous blocks. Since one “jump” arc is needed to account for each of the $3n$ discontinuities between these blocks, we set the costs of these arcs to one third the cost of a block move. This model is not only used for the A* estimate in the previous section, but also for alignment itself.⁵⁵

⁵⁵ When generating confusion networks with this model, we modify the machine to ensure there are as many insertions or substitutions as there are words in the hypothesis. In our experiments with ITGs, this cheaper model is used when we are aligning confusion networks longer than 150 bins.

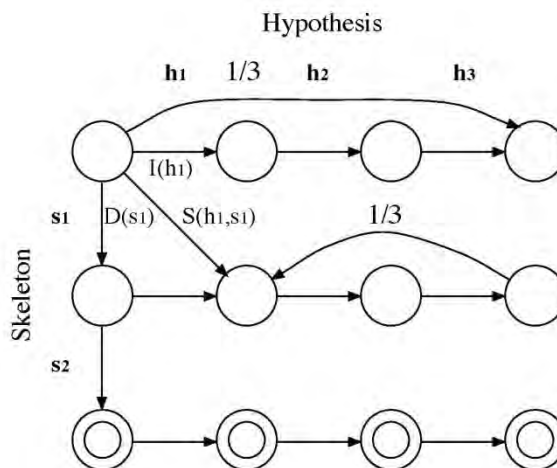


Figure 2.69: Finite state alignment model. The basic edit distance machine has been augmented with “jump” arcs which model the discontinuities introduced by block moves. While only two are pictured for space, there are jump arcs to and from each state in a row.

Oracle Computation with ITGs

By setting the cost function between a word and a confusion network bin equal to a 0/1 value (i.e., depending on whether the word is in the bin or not) we were able to easily compute the oracle invWER in a path using our ITG code: once an alignment between a confusion network and the reference sentence is obtained through a permutation of bin blocks, computing the TER simply entails counting the number of bins, which do not contain the aligned reference word, plus the number of block shifts.

This computation of the oracle path in a confusion network is much more efficient than through the Dyna code used for the oracle experiments of Karakos *et al.* (2008); the Dyna code is more general, though, because of its applicability to any kind of lattice. Oracle results appear in Section 2.8.2.4.3.

2.8.2.4.3 Experimental Setup and Results

We report results on three sets of system combination experiments: (i) a controlled comparative study of the alignment procedures described above in terms of combination performance on Arabic speech, (ii) a large-scale experiment with the ITG-based combination applied to Arabic speech (automatic recognition output) and text (newswire documents and weblogs), and (iii) a similar ITG-based experiment on Chinese speech and text. In the rest of this section we will refer to these as experiment 1, 2 and 3, respectively.

Because GALE measures performance on a per-document basis, the TER, BLEU results reported here are document averages. Furthermore, to better understand the impact of the combination on the performance of the worst translated documents, we also report “tail” results, such as

$$\text{tailBLEU} \triangleq \frac{1}{|W|} \sum_{d \in W} \text{BLEU}(d)$$

where W is a window of documents “around” the GALE target consistency level (e.g., 90% for Arabic Newswire). In more mathematical terms, W is defined as

$$W = \left\{ d: \left| \text{rank}(d) - \left\lceil \frac{\text{consistency}}{100} \times N \right\rceil \right| \leq t \right\}$$

where N is the total number of documents, $\text{rank}(d)$ is the rank of document d (with the best document having rank 1) and t is a threshold chosen so that $|W|$ is equal to a desired value. The value of $|W|$ for each genre/condition was set to 11. Tail TER is defined similarly.

The corpora used are the following:

1. **Development sets:** These are used to tune the parameters of the LM rescoring, that is, the mixture weight for the confusion network and LM and the NULL transition penalty. For experiment 1, the development set is part of Arabic MT06, consisting of roughly 260 segments (utterances) of speech. For experiment 2, the development set is Arabic DEV-07, that was made available under Phase 2 of the GALE program, consisting of roughly 580 segments of newswire text, 650 segments of weblog text, 620 utterances of broadcast news and 260 utterances of broadcast conversations. For experiment 3, the development set is the Chinese DEV-07 Blind set, containing 270 segments of newswire text, 300 segments of weblog text, 320 utterances of broadcast news and 680 utterances of broadcast conversations.
2. **Test sets:** For experiment 1, the test set is Arabic DEV-07 (speech part) described above. For experiments 2 and 3, the test set is part of the development set under Phase 3 that was made available under the GALE program, combined with the test set of the 2008 NIST MT evaluation. We refer to this set as DEV+MT08. For Arabic (experiment 2), this set consists of roughly 800 segments of newswire text, 500 segments of weblog text, 600 utterances of broadcast news and 500 utterances of broadcast conversations. For Chinese, (experiment 3), it contains 690 segments of newswire text, 660 segments of weblog text and 1480 utterances of combined broadcast news and broadcast conversations.
3. **Language models:** We experimented with two fixed (non-adaptive) language models in the rescoring of the confusion networks: one trained on roughly 180 million words of the English side of the parallel data used in GALE for training the Arabic systems (used in experiment 1) and one which also included the AFP and Xinhua sources of the English Gigaword (used in experiments 2 and 3). The models were standard 5-gram LMs using modified Kneser-Ney smoothing.

All of the above datasets have four references, except the speech portions of Chinese and Arabic DEV-07, which have one reference.

In the case of speech, an ASR system by IBM was used to generate the automatic transcriptions that were subsequently translated by the three systems. The ASR system had a cross-adapted architecture between unvoiced and voiced speaker-adaptive trained (SAT) acoustic models. The distinction between the two comes from the explicit modeling of short vowels, which are pronounced in Arabic, but almost never transcribed.

Both sets of models were trained discriminatively on approximately 500 hours of supervised data and 2000 hours of unsupervised data.

Nine MT systems were made available for the combination, seven of which were used for Experiment 1. Three of these systems were hierarchical, following the work of Chiang (2005), one was based on the Direct Translation System of Ittycheriah and Roukos (2007), one was example-based (Brown 1996), three were phrase-based statistical MT systems and one was rule-based.

The hypothesis selection technique of Hildebrand and Vogel (2008) was applied to the above systems and its output was made available to us as a separate system. In all experiments, we used it as the skeleton in our combination.

Results

We show results on the two sets of experiments.

Experiment 1

	Broadcast News					Broadcast Conversation				
	Average		Tail		Oracle	Average		Tail		Oracle
	TER	BLEU	TER	BLEU	TER	TER	BLEU	TER	BLEU	TER
Best system	48.62	30.44	55.73	23.66	-	53.86	25.14	58.60	20.01	-
Hyp. Selection	47.53	32.47	55.26	25.57	-	53.05	27.03	57.48	21.97	-
TERCOM (0/1)	46.30	33.73	53.86	26.71	24.17	51.72	27.28	56.73	22.04	30.16
TERCOM	46.23	33.27	53.89	26.16	23.81	51.66	27.40	56.59	21.95	29.69
ITG-based	46.26	33.37	53.63	26.06	23.79	51.71	27.36	56.73	21.98	29.66
Finite State	46.50	32.93	54.24	25.98	25.84	52.25	26.75	57.23	21.42	31.86
IHMM-based	46.51	33.54	54.38	26.49	27.25	51.94	27.63	56.67	22.36	33.50

Table 2.96: Combination results for Arabic ASR to English translation on DEV07 speech (Experiment 1). The best result in each column and any results which are not significantly different from it ($p = 0.05$) are shown in **bold**. Statistical significance was not measured for the “tail”, since the documents included in the tail will vary by system.

In Table 2.96, we report the average and tail TER, BLEU for two genres (broadcast news and broadcast conversations), for the following setups: (i) TERCOM-based combination using the 0/1 cost function of Rosti *et al.* (2008) (ii) TERCOM-based combination using the bin cost of Equation 2.87; (iii) ITG-based combination using Equation 2.87; (iv) finite-state reordering as described in Section 2.8.2.4.2; and (v) incremental IHMMbased combination, using the settings mentioned by He *et al.* (2008) (but excluding the semantic similarity). For the TERCOM and ITG combinations, the order with which the systems were incrementally aligned to the skeleton was determined by the edit distance from the incremental confusion network, as described above. For comparison, the results obtained with only the hypothesis selection method of Hildebrand and Vogel (2008) are also mentioned. The “Oracle” TER column shows the value of the minimum-invWER path (as computed by our ITG code) in the confusion networks.

Experiments 2 and 3

Table 2.97 and Table 2.98 show average and tail results for the four genres of both languages: newswire and weblogs (text), broadcast news and broadcast conversations (ASR). As can be seen from these results, in all conditions, the combination gives an improvement of 2-4% points compared to the best individual system.

	NW Average		NW Tail		BN Average		BN Tail	
	TER	BLEU	TER	BLEU	TER	BLEU	TER	BLEU
Best system	39.01	54.60	45.39	45.05	39.73	51.85	45.06	42.26
Hyp. Selection	39.29	56.56	48.85	46.23	37.70	56.02	43.80	47.05
ITG-based	37.49	57.81	45.23	47.20	37.30	56.36	42.43	48.01
	WB Average		WB Tail		BC Average		BC Tail	
	TER	BLEU	TER	BLEU	TER	BLEU	TER	BLEU
Best system	49.70	38.02	54.95	30.12	40.78	47.15	44.06	42.58
Hyp. Selection	49.28	41.08	53.01	34.86	41.78	49.64	44.40	45.58
ITG-based	46.70	41.86	51.38	35.40	39.26	51.20	42.75	47.09

Table 2.97: Combination results for Arabic text and ASR translation to English on DEV+MT08 (Experiment 2). The best result in each column is shown in **bold**.

	NW Average		NW Tail		BN Average		BN Tail	
	TER	BLEU	TER	BLEU	TER	BLEU	TER	BLEU
Best system	56.01	31.37	64.37	19.09	56.86	30.36	61.87	24.27
Hyp. Selection	55.67	34.46	64.17	23.26	56.23	32.39	60.82	26.43
ITG-based	52.56	36.16	61.13	23.16	53.15	33.61	56.85	27.28
	WB Average		WB Tail		BC Average		BC Tail	
	TER	BLEU	TER	BLEU	TER	BLEU	TER	BLEU
Best system	58.69	23.21	61.44	18.26	61.00	24.10	63.77	19.46
Hyp. Selection	59.99	25.50	62.99	20.86	62.56	26.83	65.30	22.22
ITG-based	57.15	26.88	60.53	21.46	58.69	28.40	61.23	24.18

Table 2.98: Combination results for Chinese text and ASR translation to English on DEV+MT08 (Experiment 3). The best result in each column is shown in **bold**.

2.8.2.5 Conclusions

Confusion network decoding may be used to combine outputs from multiple MT systems. The combination output was shown to yield significantly better TER and BLEU scores than the output of any individual MT system included in the combination. Three alignment algorithms based on the translation edit rate and how they are used in building confusion networks were first presented by the Agile team. The incremental alignment algorithms were shown to outperform the pair-wise TER alignment. There does not seem to be a significant improvement from using the flexible alignment. The results have been mixed on other test sets too; for example, the system combination experiments presented at the Fourth Workshop on Statistical Machine Translation (Rosti *et al.* 2009). The choice of the skeleton determines the word order of the output and may also help in choosing a network with fewer alignment errors. The best results were obtained by allowing all 1-best system outputs act as skeletons and assigning each confusion network a prior probability estimated from the expected alignment statistics.

Different edit costs may influence the alignment quality in the TER based alignment algorithms. Finding optimal edit costs may be time consuming as the outputs have to be realigned every time the costs change. Faster implementation of the alignment and more computing capacity may allow optimization of these edit costs. Direct optimization of the system combination weights without relying on N -best lists may also be helpful.

The NIGHTINGALE team compared two established approaches used in by the team to align translation hypotheses for the word-level MT system combination. The first method is a statistical alignment algorithm that learns alignment connections on a whole corpus of parallel system translations. The second method is a two-pass approach that is based on translation edit rate and explicitly considers synonyms. On the GALE text and audio data, we showed that both approaches result in highly significant translation quality improvements when combining up to eight state-of-the-art MT systems. A notable improvement can be also obtained when combining three relatively similar systems.

The Rosetta team showed that although previous approaches to system combination use TERCOM to align hypotheses, invWER is a more accurate approximation to TER (Table 2.95). The team also investigated a number of novel system combination methods for machine translation based on confusion network construction. Specifically, they compared confusion networks constructed using TERCOM-based alignments, ITG-based alignments and Indirect-HMM-based alignments. The results on Arabic text and speech demonstrated that combination methods almost always result in significant improvements compared to the best system output with gains of up to 4% absolute.

2.8.3. Serial System Combination for Integrating Rule-Based and Statistical Machine Translation

Authors: Roland Kuhn, Jean Senellart, Jeff Ma, Antti-Veikko Rosti, Rabih Zbib, Achraf Chalabi, Loïc Dugast, George Foster, John Makhoul, Spyros Matsoukas, Evgeny Matusov, Hazem Nader, Rami Safadi, Richard Schwartz, Jens Stephan, Nicola Ueffing and Jin Yang

2.8.3.1 Introduction

With the recent remarkable success of statistical machine translation (SMT) systems, the question arises: how can the linguistic knowledge locked up in older, rule-based machine translation (RBMT) systems most conveniently be incorporated in these systems? In many cases, RBMT systems represent an investment of several man-years of applied expertise; even if one is an ardent proponent of the SMT approach, it makes sense to capitalize on this investment.

Inside the GALE project, two teams looked at the problem of incorporating RBMT systems into SMT ones. Surprisingly, though the teams were working completely independently and on two different language pairs (Chinese-English and Arabic-English), they reached almost identical conclusions as to the best approach to pursue and attained remarkably good experimental results with this approach.

Figure 2.70 illustrates the difference between two ways of combining MT systems: parallel system combination and serial system combination. In parallel system combination, each MT system independently generates one or more translations of a source-language sentence into the target language. The target-language outputs are then

combined in some way to generate the final target-language output. Serial system combination works in a less intuitively obvious way. A first MT system generates one or more initial translations of the source-language input into the target language. The second MT system receives these target-language translations as input and generates new target-language output from them. This second system is called the “automatic post-editor”. Thus, in serial system combination the automatic post-editor “translates” target-language input to target-language output. The automatic post-editor is typically a statistical machine translation (SMT) system.

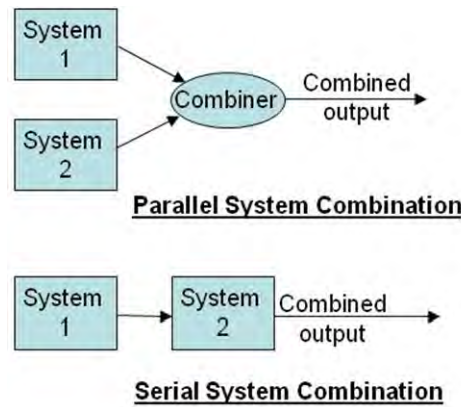


Figure 2.70: Parallel and Serial System Combination

Serial system combination was first tried in a setting where human post-editors had corrected a large quantity of translations generated by an RBMT system (Simard *et al.* 2007a). Using the RBMT system as the first system for serial combination, these researchers trained the automatic post-editor on translations from the RBMT system aligned with manually corrected versions of these same translations. The rationale is that the automatic post-editor learns how to correct certain systematic errors made by the RBMT system. In the work of Simard *et al.* (2007a), this approach was shown to produce large improvements in translation performance for the English-French language pair (both directions) compared to both to the RBMT system and the SMT system alone. Moreover, it yielded a large reduction in human post-editing effort.

A less obvious setting for serial system combination is that in which no corrections of the output of the first MT system are available. Instead, one trains the automatic post-editor on translations from the first MT system aligned with independently generated human reference translations of the same source sentences. In this setting, it is much less obvious that the automatic post-editor will learn useful rules for transforming the output of the first MT system. However, Simard *et al.* (2007b) carried out SMT Workshop experiments in which this approach was very successful. In these English-French (both directions) experiments the serial system combination consistently outperformed the pure RBMT system (SYSTRAN). The serial combination ties with the pure SMT system (PORTAGE) on a domain with large amounts of training data but outperforms it on a small domain. These results suggest that with little training data, serial combination of

RBMT and SMT is the best strategy, while with more training data, either serial combination or the use of a pure SMT system yields good results. Finally, human evaluation results for this workshop (Callison-Burch *et al.* 2007) seem to show that humans have an even higher opinion of the serial system combination than its BLEU scores suggest.

Inside the GALE project's NIGHTINGALE team, researchers at Systran and the National Research Council of Canada (NRC) put together a serial combination in which Systran's RBMT system translates Chinese to English and the NRC's system (PORTAGE) then "translates" SYSTRAN English into even better English. This serial combination yielded acceptable BLEU and TER scores compared to those of the other MT systems participating in NIGHTINGALE. However, the really interesting results were obtained when the serial combination participated in a parallel combination of five Nightingale MT systems organized by RWTH. Addition of the serial system yielded the biggest improvement obtained for any of the component systems in the parallel combination (about 0.9 BLEU). By contrast, adding SYSTRAN (by itself, not in a serial combination) to the parallel combination yielded no improvement and adding pure PORTAGE yielded only a modest improvement. It appears that the RBMT (in this case, SYSTRAN) contains valuable information that is only accessible to the parallel system combination if the RBMT's output has undergone automatic post-editing by an SMT.

Similar results were obtained by members of the Agile team inside GALE. They created a serial combination (called by them a "cascade system") in which Arabic text was translated by the Sakhr RBMT into an initial English version, which was then translated into better English by BBN's HierDec SMT system. Again, the performance of this cascade was quite good (considerably better than that of the Sakhr system on its own). Just as with the NIGHTINGALE experiments described in the previous paragraph, however, the most interesting results were obtained when the cascade system was put into a parallel system combination, containing eight systems in all. Here, inclusion of the cascade system in the combination yielded an improvement of around 1.0 BLEU.

The Agile work went further than the NIGHTINGALE work in one respect. They also tried a new version of serial combination they call "Noun-Phrase Based Combination" (NP-comb). In this version, the Sakhr RBMT system also produces an initial translation from Arabic into English; the boundaries of noun phrases (NPs) in the Arabic source and the English translation are preserved. In a second step, each of the Arabic NPs is translated by an SMT system (BBN's HierDec system). A word lattice is constructed for each input sentence, with the non-NP backbone of the sentence being provided by the RBMT system and the NP portions by the SMT system. A set of features that includes a language model score is used to find the best path through the lattice. As with the other serial system combinations described above, NP-comb yielded better scores than the pure RBMT system, but not (in general) better scores than the SMT system in the combination. However, just as in the other cases, the serial combination (here, NP-comb) made a remarkable contribution to a parallel combination. In the experiments by these Agile researchers, addition of NP-comb to a parallel system combination yielded more than 1 BLEU point improvement for Web data (no improvement was seen for newswire data).

We believe that the experiments described here are among the most interesting that resulted from the GALE project. Carried out by two completely different teams on two different language pairs, they show that the information carried by RBMT systems can be exploited within the SMT framework. The best way to do this is to build a serial combination made of RBMT followed by SMT. If desired, the serial combination system can then be incorporated in a parallel combination with other systems. The experiments reported here show that the information contributed by the serial combination is often extremely valuable (presumably, because it is complementary to information contributed by the other components).

2.8.3.2 Chinese-English Experiments

This section describes experiments carried out in the context of both the GALE project and the 2008 NIST Chinese-to-English MT evaluation. The system that generated the initial translations was SYSTRAN and the automatic post-editor was PORTAGE. The integration of the two components of the serial combination was deepened by passing confidence information from SYSTRAN on to PORTAGE.

The SYSTRAN Chinese-English system is described extensively by Yang *et al.* (2003). In particular, it uses a rule-based word-boundary module and a bilingual lexicon with about 1.2M entries, containing words, expressions and rules. This system is specifically geared to translation in the domain of science and technology. Recent evolution of this system has focused on developing structured markup making it possible to monitor rule application and to interact with the rule engine (Attnäs *et al.* 2005).

The PORTAGE system is a standard phrase based system that carries out beam search decoding using a loglinear model. Major features in this loglinear model include phrase tables derived from symmetrized IBM2 and HMM word alignments (with a phrase length limit of seven words) and a static 4-gram language model (LM) trained on the Gigaword corpus using the SRILM toolkit (Stolcke 2002). An important feature is an adapted 4-gram LM derived from the parallel corpus using the technique of Foster and Kuhn (2007); it is a linear mixture of N -gram models trained on parallel sub-corpora (Hong Kong Hansard, FBIS, UN, *etc.*). Other features are word count and phrase-displacement distortion. Parameter tuning of this loglinear combination is performed using Och's max-BLEU algorithm (Och 2003) with a closest match brevity penalty. Decoding uses the cube-pruning algorithm of Huang and Chiang (2007) with a seven-word distortion limit.

2.8.3.2.1 GALE System Combination

Recent work on parallel system combination includes that of Matusov *et al.* (2006, 2009) and Rosti *et al.* (2007a). This approach to system combination was applied to Chinese-English data available within the DARPA-sponsored GALE project by researchers at RWTH (Matusov *et al.* 2009). One of the systems in the RWTH parallel system combination is itself a combination: our hybrid SYSTRAN→PORTAGE system. This version of the hybrid system had loosely coupled components (confidence information isn't output by SYSTRAN). The other systems in this parallel combination were all pure SMT systems. In earlier experiments, an attempt had been made to include

SYSTRAN by itself, but it always received a negligible weight in the parallel system combination.

Although the SYSTRAN→PORTAGE hybrid system is not the best among the systems combined, as measured either by BLEU score or TER score, it is often assigned the heaviest weight by the system combination algorithm. Since the heavy weights assigned to the SYSTRAN→PORTAGE system could be dismissed as an artifact, experiments were also done (using 554 newswire sentences as test data) in which one system at a time was dropped from the five-way system combination. The results are shown in Table 2.99 for case-sensitive BLEU and TER (high BLEU and low TER are desirable). Here “PORTAGE” denotes a “direct” Chinese-English system from NRC, “HKUST” denotes the system provided by the Hong Kong University of Science and Technology, “RWTH PB” denotes Aachen University’s phrase-based system and “SRI” denotes the system provided by Stanford Research Institute.

Description	BLEU	TER
All (system comb).	17.0	65.4
drop PORTAGE	17.0	64.9
drop HKUST	17.2	65.0
drop RWTH PB	17.2	65.6
drop SRI	16.6	65.4
drop SYSTRAN→PORTAGE	16.1	68.0

Table 2.99: “drop-one” in GALE system combination.

When SYSTRAN→PORTAGE is dropped, the system combination performs worse by 0.9 BLEU and 2.6 TER. This performance drop is worse than that caused by dropping any other single system. One may infer that despite its unremarkable BLEU and TER scores taken in isolation (typically, these are close to the median scores of the other systems in the combination), the SYSTRAN→PORTAGE serial combination provides information that complements that of the other systems (which were all SMT systems).

2.8.3.2.2 Participation in NIST MT 2008

Both the pure statistical system PORTAGE and the serial combination SYSTRAN→PORTAGE participated in the 2008 NIST Chinese-English MT evaluation. In the serial combination, the SYSTRAN component passed confidence information on to the PORTAGE component. Both systems were trained on data from the constrained data track (thus, to create the SYSTRAN→PORTAGE system all the Chinese constrained-track training data was translated by SYSTRAN and the English output aligned with the English references was then used to train PORTAGE). Both systems placed roughly in the middle of competing systems according to BLEU score, with PORTAGE obtaining a score of 24.6 and SYSTRAN→PORTAGE obtaining a score of 25.2. (Internal tests of SYSTRAN alone on the same data give 18.2). However, SYSTRAN→PORTAGE excelled according to an important human evaluation metric. In cases where human evaluators had ranked a translated sentence sufficiently highly, they were asked an additional question as to whether the translation captured all the needed information.

SYSTRAN→PORTAGE obtained the highest number of “yes” answers among all 14 Chinese-English systems evaluated: 35.82% of all sentences.

The mechanism by which confidence information is passed from the SYSTRAN to the PORTAGE component is of interest. The output of the SYSTRAN component is broken into annotated “chunks”; some of these chunks have confidence values assigned to them. We experimented with different methods of transmitting the confidence information; for details, see the work of Ueffing *et al.* (2008). In each of these methods, a source chunk is assigned a translation into the target language and a probability for this translation. The SMT decoder hypothesizes both the translation provided in the RBMT output and translations provided by the phrase tables. The translation yielding the highest overall sentence score will be the one chosen. In the method we decided to use for the NIST evaluation, confidence values for chunks output by the SYSTRAN system are plugged directly into PORTAGE’s phrase table. Table 2.100 shows the results of this method as assessed on NIST test data. We ran a 1000-fold pair-wise BLEU bootstrap comparison, which indicates that the with-confidence version is better at the 95% confidence level for both the NIST04 and NIST08 data sets. Thus, transmission of confidence information from the SYSTRAN component to the PORTAGE component improves the hybrid system significantly.

Description	NIST04	NIST08
Baseline	34.0	25.0
System with confidence information	35.0	25.8

Table 2.100: Translation quality (BLEU score)

In addition to the automatic evaluation, we had an English native speaker manually assess the translations of the first 100 sentences of the 2004 NIST test set for the baseline system and the system with confidence information. Outputs were shuffled for each sentence to get reliable blind evaluation of the two systems. Fluency and adequacy were judged on a scale of 0-5 (0=worst, 5=best). The results are shown in Table 2.101 and they support the outcome of the automatic evaluation: the system which uses confidence is significantly preferred to the baseline. For instance, in a third of all cases for both measures, the system with confidence is judged better than the baseline system.

	Fluency	Adequacy
Baseline	1.9	2.2
System with confidence	2.1 (+10%)	2.5 (+13%)
Comparison baseline vs. system with conf: worse/equal/better	10/58/32	6/64/30

Table 2.101: NIST04 – Qualitative human evaluation

2.8.3.3 Discussion of Chinese-English Experiments

In the work presented above, RBMT→SMT serial system combination typically yielded much better BLEU scores than those obtained by the RBMT component on its own and slightly better scores than the SMT component on its own; it was awarded much

better scores by human evaluators than either RBMT or SMT on its own. According to one important human metric, the serial combination described here was ranked first of 14 systems participating in the 2008 Chinese-English NIST evaluation. Furthermore, serial combination seems to contribute valuable complementary information in a parallel system combination where the other systems are pure SMT ones.

Why does a serial combination of the form RBMT→SMT yield superior translations to those produced by either component on its own? Our perception is that the syntactic rules in the RBMT component preserve syntactic structure and allow long-distance movement of constituents. However, the RBMT component has a weak model of the target language, yielding disfluencies. Post-editing by the SMT component preserves syntactically motivated rearrangements while correcting disfluencies in the output and making more appropriate lexical choices. We have observed this behavior in several examples, such as the one shown in Table 2.102.

Description	Sentence
Chinese	欧盟官员现在最担心的是即将从非洲飞回的候鸟。
Systran	What the EU officials most were worried now is soon the migratory bird which flies back from Africa.
PORTAGE	EU officials are most concerned about is coming from Africa flew back to the migratory birds.
Systran→ PORTAGE	The EU officials are most worried about the migratory birds that fly back from Africa.

Table 2.102: three translations of a Chinese sentence.

In theory, the serial combination technique has serious limitations; it can only cumulate information loss coming from both systems. Though the SMT system may benefit from the RBMT system's syntactic analysis of the source sentence, the gain is indirect, since the SMT system is not directly informed of this analysis - this situation is not optimal. We can see the combination of the two engines as cooperation between two expert agents; the more information they exchange about their analysis of the problem, the better their combined result. The improvement gained in NIST08 by the passing of confidence information from the RBMT system to the SMT system demonstrated the potential for further improvements in combination techniques.

We can classify the problems in the current combination technique as follows: A. many2ones, B. lack of access by SMT to the RBMT lattice, C. morphology, D. constituent boundaries, as shown in Figure 2.71.

A. Many2ones

“Many2ones” are classes of source words having a common translation through RBMT. The SMT engine will see the class as a single word: having *less information* than the RBMT engine or a direct SMT engine for properly retranslating/post-editing these words. For French to English, we have seen many examples of these many2ones that are prepositions, but nouns can also be involved in many2ones.

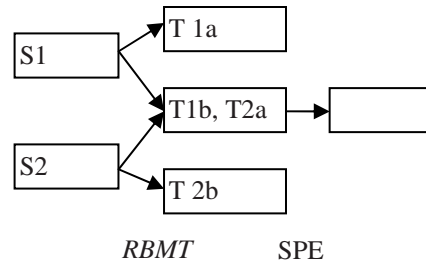


Figure 2.71: Typical example of many2one : words S1 and S2 share a common meaning. Without any context, the SMT engine cannot “correct” the RBMT output.

Not all many2ones are equally critical. For instance, for most prepositions, the context will allow the SMT engine to recapture the difference between the different source words. To deal with these words, we can allow the RBMT engine to annotate these words with their source equivalent. For instance, for French to English, the RBMT engine could annotate occurrences of the word “in” in its English output with the French word each was translated from: *e.g.*, “in.à Paris”, “in.dans Paris”. Annotation with the source word should be performed selectively on frequent target words; if it is performed on rarer words, it may interfere with word alignment and phrase table building for the SMT system.

B. Lack of Access by SMT to the RBMT Lattice

In the serial combination technique described previously, a single RBMT translation is passed to an SMT engine, which carries out post-editing. Post-editing involves considering alternative lexical choices (by phrase substitution) and alternative word orders, letting a language model decide on the best hypothesis. Human evaluation of the type of substitution performed by this mechanism shows that an important part of these substitutions is the choice of an appropriate meaning, or the choice of preposition or right determiner. In most cases, most of these alternatives were already known to the RBMT engine. Thus, the SMT engine performs an unnecessarily complicated task, reconstructing from a single RBMT translation a set of alternative constructions and meanings for a given source sentence that were originally available inside the RBMT system.

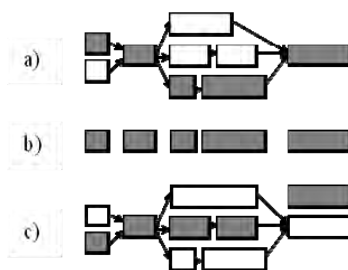


Figure 2.72: a) Represents the internal choices in the RBMT generation module, b) the unique solution selected, c) the option lattice rebuilt by the SMT engine – most of the original generation alternatives are rebuilt.

Direct use of this RBMT lattice by the SMT system is, however, not simple. Our current approach is to apply a first lattice decoding in the last phase of RBMT output to produce N -best translations and to use this N -best list as an input to SMT decoding.

C. Morphology

RBMT systems are much better at handling morphology than SMT systems are. In the serial combination setup described previously, the SMT engine sometimes degrades morphological choices or gender/case agreement while improving meaning selection. Another problem is that when translating into a language with a richer morphology than the source language, the larger vocabulary of the target language dilutes the information contained in the SMT models, causing the SMT in the serial combination setup to perform less well than an SMT trained to translate directly from the source language. A possible workaround would be to let linguistically motivated rules select valid sequences in the SMT N -best list or to postpone morphology generation to a further stage of the process.

D. Constituent Boundaries

One particularly frustrating class of post-editing error observed in a typical serial combination occurs when the SMT component carries out word reordering to improve fluency without respecting the logic of constituents as analyzed by the RBMT engine. While some reordering (most often restricted to local reordering) carried out by the SMT component is valid and corrects invalid word order choices made by the RBMT component, erroneous transformations can break the desired sentence structure and may lead to very bad mistranslations. One could argue that the quality of the constituents recognized by RBMT might not be good enough to rely on. It seems likely that one could get performance improvements for serial combination by constraining the reordering carried out by the SMT component to respect the constituent boundaries defined by the RBMT component.

In our future work, we plan to explore deeper integration of the two systems in the serial combination. The work we have already done on passing confidence information from the first system to the second shows that this approach can yield important dividends. We believe that it will be possible to attain a deeper integration of the two

tools, which will allow each of them to constrain the other and allow both of them to operate simultaneously instead of in sequence.

2.8.3.4 Arabic-English Experiments

In this section we describe in detail experiments conducted by the Agile team on the Arabic-English serial combination, which we refer to as cascade experiments. As mentioned above, these experiments use the Sakhr Arabic-English translation system for the rule-based component and BBN's HierDec system (Shen *et al.* 2008) for the statistical component. The BBN system is trained on 53 million words of Arabic-English parallel data from the GALE training corpus. The baseline SMT system to which the cascade results are compared is also the BBN HierDec system trained on the same data. A 3-gram language model is used for decoding and 5-gram language model is used for N-best rescoring for both setups. Both systems were tuned to maximize IBM BLEU.

The data sets on which the experiments we run are composed of subsets of NIST MT evaluation and GALE development and evaluation data sets. For newswire data, the set Test.nw consists of 1194 segments, SysCombTune.nw consists of 2118 segments, and Test.nw consists of 1031 segments. For the web data, the set Tune.web contains 3278 segments, SysCombTune.web contains 3132 segments, and Test.web 1293 segments. It is worth noting that when a certain data set is divided, the division always respects document boundaries, so that no two sentences from the same document end in two different sets.

For the Sakhr-BBN cascade combination experiments, the system parameter weights were tuned on Tune.nw. The two sets SysCombTune.nw and Test.nw were used as test sets. The scores from these experiments are shown in Table 2.103. These results are for lower case output. The cascade system scores show a significant improvement over the scores of the Sakhr system for newswire. They have about 10 points lower in TER and 15 points higher in IBM BLEU points. This is consistent with the results obtained by the NIGHTINGALE team on Chinese-English, which are described in Section 2.8.3. In this case also, it appears that the SMT system is able to learn useful post-editing rules that make the output of the RBMT system better. The use of a language model in the SMT component of the cascade architecture contributes significantly to obtaining a more fluent output. It should also be noted that the scores of the cascade system are slightly lower than those of the baseline HierDec data trained on the same data.

newswire	Tune.nw		SysCombTune.nw		Test.nw	
	TER	BLEU	TER	BLEU	TER	BLEU
Sakhr RBMT	44.22	39.69	48.73	38.38	49.52	38.29
BBN Hierdec	38.56	55.79	39.24	53.52	40.17	51.64
Cascade System	38.31	55.53	38.84	53.94	40.25	51.56

Table 2.103: Results of cascade combination of Sakhr RBMT and BBN HierDec systems for Arabic newswire

newswire	Tune.nw			Test.nw		
	TER	BLEU	METEOR	TER	BLEU	METEOR
With Cascade Output	34.71	60.09	71.41	35.65	57.86	70.21
Without Cascade Output	34.21	61.10	71.86	35.23	58.85	70.57

Table 2.104: Results of word-level system combination of 8 MT systems, with and without the output of the Sakhr+BBN cascade system for newswire data.

web	Tune.nw			Test.nw		
	TER	BLEU	METEOR	TER	BLEU	METEOR
With Cascade Output	54.65	28.86	53.37	46.20	45.50	61.82
Without Cascade Output	54.13	30.05	54.35	45.81	46.42	62.78

Table 2.105: Results of word-level system combination of 8 MT systems, with and without the output of the Sakhr+BBN cascade system for web data.

Next, we ran experiments to measure the effect of adding the output of the Sakhr-BBN cascade system as an additional system to parallel word-level system combination. The baseline, shown in the first row of Table 2.104 and Table 2.105, consists of the combination of the outputs of 8 different systems from the GALE Agile team. These include the BBN HierDec system and the Sakhr rule-based system, in addition to six other statistical systems that use a variety of translation models, including phrase based and syntactic models. The outputs of the systems are combined at the word level by building a word lattice from the aligned outputs of the different systems and searching for the best path through that lattice using language model scores and other features. The details of this method can be found in the work of Rosti *et al.* (2007a and 2008). To determine whether there is a benefit from adding the output of the Sakhr-BBN cascade system, we performed another parallel system combination experiment in which the output of the cascade system is also added as another system. In this case, we tune on the sets SysCombTune.nw and SysCombTune.web and test on Test.nw and Test.web respectively. The results in Table 2.104 and Table 2.105 show that there is an improvement of about one BLEU point on both Test.nw and Test.web. The improvement suggests that the cascade system produces an output that is significantly different from the two baseline output and which the parallel combination benefits from.

2.8.3.5 Noun Phrase Based Combination

2.8.3.5.1 Procedure

The method described in this section is also inspired by the insight that RBMT systems are more likely to produce a correct sentence structure, while SMT systems are more likely to correctly translate phrases that the RBMT system would translate literally. This is especially true when the SMT system uses a phrase-based model.

In this method, the Arabic source is translated using the Sakhr rule-based system. Two outputs are then produced: an English translation of that sentence and indices of the boundaries for each noun phrase in the Arabic source, and the indices of the corresponding translation for that noun phrase in the English output. Indices for nested

noun phrases are also produced together with those of the containing noun phrases. The mapping to the corresponding indices on the target side can be also obtained in a straightforward manner, since the translation process is deterministic. Noun phrases are extracted from the Arabic source and translated through an SMT system (BBN's HierDec), then a word lattice is constructed for each input sentence, consisting of:

1. A base chain of arcs containing the base noun phrases (i.e., noun phrases that don't contain other noun phrases), as well as non-NP spans in the original sentence, such that there are non-consecutive noun phrase arcs.
2. For each non-base noun phrase, an arc that spans that noun phrase is added to the lattice, such that the edges of the added arc coincide with the start and end nodes of the span in the base chain constructed in 1.
3. For each noun phrase in the lattice (both base and non-base), a set of parallel arcs is added. The arcs are labeled with the N-best SMT translations of the noun phrase.
4. Each arc is labeled with the following features in addition to the translation output words:
 - (a) SLM: The language model score.
 - (b) SMT_ARC: Feature with value 0 if the arc translation is from the RBMT system, 1 if it is from the BBN SMT system.
 - (c) W_NUM: Number of words on the arc label.
 - (d) A_NUM: Constant. This contributes to a path-level feature, whose value is the number of arcs in the path.
 - (e) IS_NP: Feature with value 1 if the arc corresponds to a noun-phrase span, 0 otherwise.
 - (f) IN NON BASE NP: Feature with value 1 if the noun phrase contains nested noun phrases, 0 otherwise (i.e., is base noun phrase).

All feature values can be straightforwardly added to the lattice when it is being built, except for the language model score (SLM). Assigning the language model properly requires the additional step of expanding the lattice (Odell 1995). Briefly, the sequence of translation words on an arc can have as many different word histories as there are incoming arcs into the starting node of that arc. In order to compute the language model score for a given arc string, the lattice has to be expanded such that the arc string has a unique N-gram history. The language model score at each new arc can now be computed based on the unique word history of that arc.

After the language model scores are computed, each link has a vector of feature scores. The total score of a given arc is the dot product of the vector of feature scores with the vector of feature weights. The path with the highest such score is selected.

The set of feature weights are tuned on a tuning data set to maximize the BLEU score. The objective function is minimized using Powell's method (Brent 1973) since it is not differentiable. The A* algorithm is used to get an N-best list of translations from the lattice. Finally, the N-best is re-ranked using a 5-gram language model.

2.8.3.5.2 Noun Phrase Based Experiments

In these experiments, we also used the BBN HierDec system (Shen *et al.* 2008), trained on 200 million words of Arabic-English parallel from the P2 GALE training data and the UN data. The same statistical system is used to translate the full segments in the standalone HierDec system and the noun phrases used in the noun phrase combination system.

The newswire data used in the experiments described in this section are from the NIST MT02-MT05 test sets. For the web data, we used two test sets, ng_y1_q4 Tune containing 2079 segments and ng_y1_q4 Test containing 2128 segments, all taken from the GALE year 1 quarter 4 LDC web parallel data. The web data has one reference, which partly explains the lower scores. All experiments were tuned to maximize the BLEU score. For newswire, we tune on NIST_02_05 Tune and test on NIST_MT02_05 Test. For the web experiments, we tune on ng_y1_q4 Tune and test on ng_y1_q4 Test. All scores are for lower case English output.

The first experiment uses the top translation produced by the HierDec system for each noun phrase. The noun phrase combination system improves over the scores of the RBMT system significantly. For the newswire test set, there is an increase of 9 BLEU points and a decrease of about 7.5 TER points, as Table 2.106 shows. But the scores remain noticeably lower than the standalone HierDec system. Table 2.107 shows the results for the web test set. In this case, noun-phrase combination also shows an improvement of 2 BLEU points and 4.5 TER points. In this case, the noun phrase combination system shows an improvement of about 0.45 points over the standalone HierDec system. The improvement in scores is due to the better translation of the noun phrases and the use of the language model in the rescoring of the lattice. We also experimented with using the top 50 translations for each NP in the lattice. This provides a small improvement over using the top translation for newswire and no improvement for the web.

newswire	NIST_MT02_05_Tune			NIST_MT02_05_Test		
	TER	BLEU	METEOR	TER	BLEU	METEOR
Sakhr RBMT	51.74	36.86	65.66	51.42	37.13	56.83
BBN Hierdec	38.73	55.42	72.02	38.95	55.48	71.44
NP Comb 1-best	44.20	45.79	69.32	44.20	46.07	69.25
NP Comb 50-best	43.45	46.80	68.87	43.48	46.92	68.97

Table 2.106: Results of noun-phrase based combination for Arabic newswire data.

web	ng_y1q4_Tune			ng_y1q4_Test		
	TER	BLEU	METEOR	TER	BLEU	METEOR
Sakhr RBMT	69.02	15.69	45.11	67.96	15.65	44.65
BBN Hierdec	60.41	19.76	47.06	61.72	17.03	44.15
NP Comb 1-best	62.93	18.98	48.07	63.39	17.48	46.10
NP Comb 50-best	62.64	18.84	47.92	63.37	17.19	45.84

Table 2.107: Results of noun phrase based combination for Arabic web data.

web	ng_y1q4_Tune			ng_y1q4_Test		
	TER	BLEU	METEOR	TER	BLEU	METEOR
RBMT+Hierdec	62.72	20.17	50.53	63.40	18.30	48.34
RBMT+Hierdec+ NPcomb	61.28	21.28	51.26	61.87	19.25	49.23
RBMT+ NPcomb	64.06	19.25	50.56	63.96	18.11	49.24
Hierdec+NPcomb	61.14	21.26	51.20	61.95	18.86	49.19

Table 2.108: Results of adding noun phrase system combination output to the word level combination of Sakhr's RBMT system and BBN's HierDec system.

As in the case of the cascade system experiments in this section, we also experimented with using the output of the noun phrase combination system in word level system combination experiments. Table 2.108 shows the results of adding the output of the noun phrase combination system (NP-comb) to outputs of the rule-based system (RBMT) and the hierarchical decoder SMT system (HierDec) for the web data. As the results show, adding the output of the noun-phrase noun phrase system results in a gain of 1.1 BLEU points over only combining the outputs of the rule based and the hierarchical SMT system. However, no similar gains were observed with the newswire data.

2.8.3.6 Conclusions

We have investigated two methods for integrating the Sakhr rule-based machine translation system and BBN statistical machine translation system with hierarchical decoding: the cascade combination method and the noun phrase combination method. We have shown that both methods improve the scores of the rule-based system considerably. And, that it improves over the score of the SMT system in the case of the cascade combination. We have also shown that the addition of the output of these systems to word-level system combinations beneficial. Future work on the cascade system can investigate additional features that incorporate the Arabic source within the statistical model of the SMT system. Additional work similar to the noun phrase combination work could target other linguistic structures and different choices for the skeleton on which the translation lattice is built.

References

- Aldous, D. 1985. Exchangeability and related topics. *École d'été de probabilités de Saint-Flour XIII-1983*, pp. 1–198. Springer: Berlin.
- Al-Onaizan, Y. 2005. IBM Arabic-to-English MT Submission. *Presentation given at DARPA/TIDES*
- Al-Onaizan, Y., J. Curin, M. Jahr, K. Knight, J. Lafferty, I. D. Melamed, N. A. Smith, F.J. Och, D. Purdy and D. Yarowsky. 1999. Statistical Machine Translation. *CLSP Research Notes* No. 42, Johns Hopkins University.
- Ando, R. K. and L. Lee. 2003. Mostly unsupervised statistical segmentation of Japanese kanji sequences. *Journal of Natural Language Engineering*, 9, pp. 127–149.
- Andrew, G. 2006. A Hybrid Markov/Semi-Markov Conditional Random Field for Sequence Segmentation. *Proceedings of EMNLP*, Sydney, Australia.
- Attnäs, M., P. Senellart and J. Senellart. 2005. Integration of SYSTRAN MT Systems in an Open Workflow. *Proceedings of MT Summit X*.
- Avramidis, E. and P. Koehn. 2008. Enriching Morphologically Poor Languages for Statistical Machine Translation. *Proceedings of ACL/HLT*, pp.763–770, Columbus, Ohio.
- Ayan, N.F., J. Zheng and W. Wang. 2008. Improving Alignments for Better Confusion Networks for Combining Machine Translation Systems. *Proceedings of Coling*, pp. 33–40.
- Badr, I., R. Zbib and J. Glass. 2008. Segmentation for English-to-Arabic Statistical Machine Translation. *Proceedings of ACL/HLT*, Short Papers, pp.153–156, Columbus, OH.
- Bahl, L.R., F. Jelinek and R.L. Mercer. 1983. A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-5(2): pp. 179–190.
- Banerjee, S. and A. Lavie. 2005. Meteor: An automatic metric for MT evaluation with improved correlation with human judgments. *Proceedings of the 43th Annual Meeting of the Association for Computational Linguistics (ACL)*.
- Bangalore, S., G. Bordel and G. Riccardi. 2001. Computing Consensus Translation from Multiple Machine Translation Systems. *Proceedings of ASRU*, pp. 351–354.
- Bellman, R. 1957. *Dynamic Programming*. Princeton University Press: Princeton, NJ.
- Bender, O., R. Zens, E. Matusov and H. Ney. 2004. Alignment Templates: the RWTH SMT System. *Proceedings of IWSLT*, pp. 79–84, Kyoto, Japan.
- Berger, A.L., V.D. Pietra and S.D. Pietra. 1996. A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1), pp. 39–71.
- Bertoldi, N., R. Zens and M. Federico. 2007. Speech Translation by Confusion Network Decoding. *Proceedings of ICASSP*, Honolulu, Hawaii.
- Bilmes, J. and K. Kirchhoff. 2003. Factored Language Models And Generalized Parallel Backoff. *Proceedings of HLT*.

- Birch, A., M. Osborne and P. Koehn. 2007. CCG Supertags in Factored Statistical Machine Translation. *Proceedings of the Second Workshop on Statistical Machine Translation*, pp. 9–16, Prague, Czech Republic.
- Blackwood, G. A. de Gispert and W. Byrne. 2008. Phrasal Segmentation Models for Statistical Machine Translation. *Proceedings of Coling*, Manchester, UK.
- Blackwood, G. A. de Gispert, J. Brunning and W. Byrne. 2008a. Large-scale Statistical Machine Translation with Weighted Finite State Transducers. *Proceedings of Finite-State Methods and Natural Language Processing*.
- Blei, D., A. Ng and M.I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3: pp. 1107-1135.
- Blunsom, P. and T. Cohn. 2006. Discriminative Word Alignment with Conditional Random Fields. *Proceedings of Coling/ACL*, pp. 65–72, Morristown, NJ.
- Blunsom, P., T. Cohn and M. Osborne. 2008. A Discriminative Latent Variable Model for Statistical Machine Translation. *Proceedings of ACL*.
- Bod, R., R. Scha and K. Sima'an, eds. 2003. *Data-Oriented Parsing*. CSLI Publications.
- Bojar, O. 2007. English-to-Czech Factored Machine Translation. *Proceedings of the Second Workshop on Statistical Machine Translation*, pp. 232–239, Prague, Czech Republic.
- Bonneau Maynard, H., A. Allauzen, D. D'echelotte and H. Schwenk. 2007. Combining morphosyntactic enriched representation with n-best reranking in statistical translation. *Proceedings of NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pp. 65–71, Rochester, NY.
- Brants, T., A. C. Popat, P. Xu, F.J. Och and J. Dean. 2007. Large Language Models in Machine Translation. *Proceedings of EMNLP-CoNLL*, pp. 858–867.
- Brent, R. 1973. *Algorithms for Minimization Without Derivatives*. Prentice-Hall.
- Brown, P. F., V.J.D Pietra, S.A.D Pietra and R.L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics* 19(2), pp. 263–311.
- Brown, R.D. 1996. Example-Based Machine Translation in the Pangloss System. *Proceedings of Coling*, pp.169-174.
- Brown, P.F., S. A. Della Pietra, V. J. Della Pietra and R. L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *Computational Linguistics* 19(2): pp. 263–311.
- Brown, P., S. Della Pietra, V. Della Pietra and R. Mercer. 1991. Word-sense Disambiguation Using Statistical Methods. *Proceedings of ACL*, pp. 264-270, Berkeley, CA.
- Brown, P.F., J. Cocke, S.A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R.L. Mercer and P.S. Roossin. 1990. A Statistical Approach to Machine Translation. *Computational Linguistics*, 16(2): pp. 79–85.

- Buckwalter, T. 2002. Buckwalter Arabic Morphological Analyzer Version 1.0. *Linguistic Data Consortium, University of Pennsylvania*. LDC Catalog No.: LDC2002L49.
- Bulyko, I., S. Matsoukas, R. Schwartz, L. Nguyen and J. Makhoul. 2007. Language Model Adaptation in Machine Translation from Speech. *Proceedings of IEEE-ICASSP*.
- Callison-Burch, C., C. Fordyce, P. Koehn, C. Monz and J. Schroeder. (Meta-) Evaluation of Machine Translation. 2007. *Proceedings of 2nd ACL Workshop on Statistical Machine Translation*, pp. 136-158.
- Callison-Burch, C., M. Osborne and P. Koehn. 2006. Re-evaluating the Role of BLEU in Machine Translation Research. *Proceedings of EACL*, pp. 249–256, Trento, Italy.
- Carpuat, M. and D. Wu. 2008. Evaluation of Context-dependent Phrasal Translation Lexicons for Statistical Machine Translation. *Proceedings of LREC*. European Language Resources Association (ELRA): Marrakech.
- Carpuat, W. and D. Wu. 2007. How Phrase Sense Disambiguation Outperforms Word Sense Disambiguation for Statistical Machine Translation. *Proceedings of the Conference on Theoretical and Methodological Issues in Machine Translation*, pp. 43-52. Skovde, Sweden.
- Carpuat, M. and D. Wu. 2007a. Improving Statistical Machine Translation using Word Sense Disambiguation. *Proceedings of EMNLP-CoNLL*, pp. 61-72. Prague, Czech Republic.
- Carpuat, M. and D. Wu. 2005. Word Sense Disambiguation vs. Statistical Machine Translation. *Proceedings of ACL*, pp.387-394. Ann Arbor, MI.
- Cettolo, M., F. Marcelllo, D. Pighin and N. Bertoldi. 2008. Shallow-syntax Phrase-Based Translation: Joint Versus Factored String-to-Chunk Models. *Proceedings of AMTA*.
- Chang, P., M. Galley and C.D. Manning. 2008. Optimizing Chinese Word Segmentation for Machine Translation Performance. *Proceedings of the Third Workshop on Statistical Machine Translation*. pp. 224-232, Columbus, Ohio.
- Chan, Y.S., H.T. Ng and D. Chiang. 2007. Word Sense Disambiguation Improves Statistical Machine Translation. *Proceedings of ACL*. Prague, Czech Republic.
- Chappelier, J. -C., and M. Rajman. (1998) A Generalized CYK Algorithm for Parsing Stochastic CFG. <ftp://ftp.inria.fr/INRIA/Projects/Atoll/TAPD98/chappelier.ps.gz>
- Charniak, E., K. Knight and K. Yamada. 2003. Syntax-based Language Models For Statistical Machine Translation. *Proceedings of MT Summit IX*.
- Charniak, E., S. Goldwater and M. Johnson. 1998. Edge-based Best-first Chart Parsing. *Proceedings of the Sixth Workshop on Very Large Corpora*, pp.127-133.
- Chen, B., M. Cettolo and M. Federico. 2006. Reordering Rules for Phrase-based Statistical Machine Translation. *Proceedings of IWSLT*. Kyoto, Japan.
- Chen, S.F. and J. Goodman. 1998. An Empirical Study of Smoothing Techniques for Language Modeling. Technical Report TR-10-98, Harvard University Center for Research in Computing Technology.

- Cheppalier, J., M. Rajman, R. Aragues and A. Rozenknop. 1999. Lattice Parsing for Speech Recognition. *Proceedings of Conference sur le Traitement Automatique du Langage Naturel*, pp. 95–104.
- Cheppalier, J.C. and M. Rajman. 1998. A Generalized CYK Algorithm for Parsing Stochastic CFG. *Proceedings of the First Workshop on Tabulation in Parsing and Deduction (TAPD98)*, pp. 133–137.
- Cherry, C. and D. Lin. 2003. A Probability Model to Improve Word Alignment. *Proceedings of ACL*, pp.88–95, Sapporo, Japan.
- Chiang, D. and D. Wu, eds. 2008. *Proceedings of Second Workshop on Syntax and Structure in Statistical Translation, at ACL/HLT*. Columbus, OH.
- Chiang, D., Y. Marton and P. Resnik. 2008. Online Large-margin Training of Syntactic and Structural Translation Features. *Proceedings of EMNLP*, pp. 224–233. Hawaii.
- Chiang, D. 2007. Hierarchical phrase based translation. *Computational Linguistics*, 33(2), pp. 201–228.
- Chiang, D. 2005. A Hierarchical Phrase-based Model for Statistical Machine Translation. *Proceedings of ACL*, pp. 263–270. Ann Arbor, MI.
- Collins, M. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. *Proceedings of EMNLP*.
- Collins, M. 2003. Head-driven Statistical Models for Natural Language Parsing. *Computational Linguistics* 29(4): pp. 589–673.
- Collins, M., P. Koehn and I. Kucerova. 2005. Clause Restructuring for Statistical Machine Translation. *Proceedings of ACL* pp.531–540. Ann Arbor, MI.
- Corston-Oliver, S. and M. Gamon. 2004. Normalizing German and English Inflectional Morphology to Improve Statistical Word Alignment. *Proceedings of AMTA*, pp. 48–57.
- Costa-Jussa, M. R. and J.A.R. Fonollosa. 2006. Statistical Machine Reordering. *Proceedings of EMNLP*, pp. 70–76, Sydney, Australia.
- Costa-Jussa, M. R. and J.A.R. Fonollosa. 2007. Analysis of Statistical and Morphological Classes to Generate Weighted Reordering Hypotheses on a Statistical Machine Translation System. *Proceedings of the Second Workshop on Statistical Machine Translation*, pp. 171–176. Prague, Czech Republic.
- Cowan, B., I. Kucerova and M. Collins. 2006. A Discriminative Model for Tree-to-Tree Translation. *Proceedings of EMNLP*, pp. 232–241. Sydney, Australia.
- Crego, J. M. and N. Habash. 2008. Using Shallow Syntax Information to Improve Word Alignment and Reordering for SMT. *Proceedings of the Third Workshop on Statistical Machine Translation*, pp. 53–61, Columbus, Ohio.
- Crego, J. M. and J.B. Marino. 2007. Syntax-enhanced Ngram-based SMT. *Proceedings of the MT Summit XI*.
- Crego, J. M., A. de Gispert, P. Lambert, M.R. Costa-Jussa, M. Khalilov, R.E. Banchs, J.B. Marino and J.A.R. Fonollosa. 2006. N-gram-based SMT System Enhanced with

- Reordering Patterns. *Proceedings of the Workshop on Statistical Machine Translation* pp.162–165. New York, NY.
- Crego, J.M. and J.B. Marino. 2006. Integration of POS-tag-based Source Reordering into SMT Decoding by an Extended Search Graph. *Proceedings of AMTA*, pp. 29–36. Cambridge, MA.
- Dagan, I. and A. Itai. 1994. Word Sense Disambiguation Using a Second Language Monolingual Corpus. *Computational Linguistics* 20(4): pp. 563-596.
- de Gispert, A., S. Virpioja, M. Kurimo and W. Byrne. 2009. Minimum Bayes Risk Combination of Translation Hypotheses From Alternative Morphological Decompositions. *Proceedings of NAACL-HLT*.
- de Gispert, A. and J.B. Marino. 2008. On the Impact of Morphology in English to Spanish Statistical MT. *Speech Communication* 50: pp. 1034–1046.
- de Gispert, A. 2006. Introducing Linguistic Knowledge into Statistical Machine Translation. *Ph.D. thesis, Dept. Signal Theory and Communications, Universitat Politècnica de Catalunya*.
- De Nero, J. and D. Klein. 2008. The Complexity of Phrase Alignment Problems. *Proceedings of ACL-08, Companion volume*, pp. 25–28. Columbus, OH.
- Della Pietra, S., V. Della Pietra and J. Lafferty. 1995. Inducing Features of Random Fields. *Technical Report, Department of Computer Science, Carnegie-Mellon University, CMU-CS-95-144*.
- Dempster, A.P., N. M. Laird and D. B. Rubin. 1977. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1): pp. 1–38.
- DeNeefe, S., K. Knight, W. Wang and D. Marcu. 2007. What Can Syntax-based MT Learn from Phrase-based MT? *Proceedings of EMNLP-CoNLL*.
- Deng, Y. and W. Byrne. 2008. HMM Word and Phrase Alignment for Statistical Machine Translation. *IEEE Transactions on Audio, Speech and Language Processing* 16(3): pp. 494–507.
- Deng, Y. and W. Byrne. 2006. MTTK: An Alignment Toolkit for Statistical Machine Translation. *Presented in the HLT-NAACL Demonstrations Program*.
- Denoual, E. 2007. Analogical Translation of Unknown Words in a Statistical Machine Translation Framework. *Proceedings of the MT Summit XI*.
- Diab, M., M. Ghoneim and N. Habash. 2007. Arabic Diacritization in the Context of Statistical Machine Translation. *Proceedings of the Machine Translation Summit*.
- Ding, Y. and M. Palmer. 2005. Machine Translation using Probabilistic Synchronous Dependency Insertion Grammars. *Proceedings of ACL*, pp. 541–548. Ann Arbor, MI.
- Dorr, B. J. 1994. Machine Translation Divergences: a Formal Description and Proposed Solution. *Computational Linguistics* 20(4).
- Dreyfus, S.E. and A.M. Law. 1977. *The Art and Theory of Dynamic Programming (Mathematics in Science and Engineering; vol. 130)*. Academic Press: New York, N.Y.

- Dugast, L., J. Senellart and P. Koehn. 2009. Statistical Post Editing and Dictionary Extraction. *Proceedings of EACL*, pp.110-114.
- Dugast, L., J. Senellart and P. Koehn. 2007. Statistical Postediting on SYSTRAN's Rule-based Translation System. *Proceedings of ACL, Workshop on Statistical Machine Translation*, pp. 220-223.
- Durbin, R., S.R. Eddy, A. Krogh and G. Mitchison. 1998 *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press.
- Durgar El-Kahlout, I. and K. Oflazer. 2006. Initial Explorations in English to Turkish Statistical Machine Translation. *Proceedings of the Workshop on Statistical Machine Translation*, pp. 7–14. New York, NY.
- Dyer, C. J., S. Muresan and P. Resnik. 2008. Generalizing Word Lattice Translation. *Proceedings of ACL/HLT* pp.1012–1020. Columbus, OH.
- Dyer, C. J. 2007a. The “Noisier Channel”: Translation from Morphologically Complex Languages. *Proceedings of the Second Workshop on Statistical Machine Translation*, pp. 207–211. Prague, Czech.
- Dyer, C. J., 2007b. The University of Maryland Translation System for IWSLT 2007. *Proceedings of IWSLT*.
- Eck, M., S. Vogel and A. Waibel. 2005. Low Cost Portability for Statistical Machine Translation Based on N-gram Coverage. *Proceedings of MT Summit X*, pp.227-234.
- Eck, M., S. Vogel and A. Waibel. 2004. Language Model Adaptation for Statistical Machine Translation Based on Information Retrieval. *Proceedings of LREC*, pp.327-330.
- Eisner, J., E. Goldlust and N.A. Smith. 2005. Compiling Comp Ling: Weighted Dynamic Programming and the Dyna Language. *Proceedings of HLT/EMNLP*, pp.281-290.
- Eisner, J. 2003. Learning Non-isomorphic Tree Mappings for Machine Translation. *Proceedings of ACL*, companion volume, pp.205–208. Sapporo, Japan.
- Eisner, J. and G. Satta. 1999. Efficient Parsing for Bilexical Context-free Grammars and Head Automaton Grammars. *Proceedings of ACL*.
- El Isbihani, A., S. Khadivi, O. Bender and H. Ney. 2006. Morpho-syntactic Arabic Preprocessing for Arabic to English Statistical Machine Translation. *Proceedings on the Workshop on Statistical Machine Translation*, pp. 15–22. New York, NY.
- Elming, J. 2008a. Syntactic Reordering Integrated with Phrasebased SMT. *Proceedings of Second Workshop on Syntax and Structure in Statistical Translation, at ACL/HLT*. pp. 46–54. Columbus, OH.
- Elming, J. 2008b. Syntactic Reordering Integrated with Phrasebased SMT. *Proceedings of Coling*, pp.209–216. Manchester, UK.
- Emerson, T. 2005. The Second International Chinese Word Segmentation Bakeoff. *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pp. 123–133.
- Erosheva, E., S. Fienberg and J. Lafferty. 2004. Mixed Membership Models of Scientific Publications. *Proceedings of the National Academy of Sciences* volume 101 of Suppl. 1.

- Evermann, G. and P.C. Woodland. 2000. Posterior Probability Decoding, Confidence Estimation and System Combination. *Proceedings of Speech Transcription Workshop*.
- Fellbaum, C. ed. 1998. *An Electronic Lexical Database*. MIT Press.
- Fiscus, J.G. 1997. A Post-Processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER). *Proceedings of ASRU*, pp.347-354.
- Foster, G. and R. Kuhn. 2007. Mixture-Model Adaptation for SMT. *Proceedings of 2nd ACL Workshop on Statistical Machine Translation*, pp. 128-135.
- Fraser, A. and D. Marcu. 2007. Measuring Word Alignment Quality for Statistical Machine Translation. *Computational Linguistics* 33(3): pp. 293–303.
- Fraser, A. and D. Marcu. 2006a. Semisupervised training for statistical word alignment. *Proceedings of ACL/Coling*, pp.769-776. Sydney, Australia.
- Fraser, A. and D. Marcu. 2006b. Measuring Word Alignment Quality for Statistical Machine Translation. *Technical report, ISI-University of Southern California*.
- Frederick, J. and R.L. Mercer. 1980. Interpolated Estimation Of Markov Source Parameters From Sparse Data. *Proceedings of Workshop on Pattern Recognition in Practice*. Amsterdam, The Netherlands.
- Freund, Y., H.S. Seung, E. Shamir and N. Tishby. 1997. Selective Sampling Using the Query by Committee Algorithm. *Machine Learning* 28(2-3): pp. 133.168.
- Fung, P., W. Zhaojun, Y. Yang and D. Wu. 2006. Automatic Learning of Chinese-English Semantic Structure Mapping. *Proceedings of IEEE/ACL Workshop on Spoken Language Technology*.
- Gale, W. and G. Sampson. 1996. Good-Turing Frequency Estimation without Tears. *Journal of Quantitative Linguistics* 2(3).
- Gale, W.A., K.W. Church and D. Yarowsky. 1992. A Method for Disambiguating Word Senses in a Large Corpus. *Computers and the Humanities* 26: pp. 415-439.
- GALE. 2008. NIST Machine Translation Evaluation for GALE Home Page. <http://www.nist.gov/speech/tests/gale>.
- Galley, M., J. Graehl, K. Knight, D. Marcu, S. DeNeefe, W. Wang and I. Thayer. 2006. Scalable Inference and Training of Context-rich Syntactic Translation Models. *Proceedings of ACL/Coling*, pp. 961–968. Sydney, Australia.
- Galley, M., M. Hopkins, K. Knight and D. Marcu. 2004. What’s in a Translation Rule? *Proceedings of HLT/NAACL* 4: pp. 273–280.
- Gao, J., M. Li, A. Wu and C. Huang. 2005. Chinese Word Segmentation and Named Entity Recognition: A Pragmatic Approach. *Computational Linguistics* 31(4).
- Gao, Q. and S. Vogel. 2008. Parallel Implementations of Word Alignment Tool. *Software Engineering, Testing and Quality Assurance for Natural Language Processing*, pp. 49–57.
- Ge, N. 2004. Improvement in Word Alignments. *Presentation given at DARPA/TIDES MT workshop*.

- Geman, S. and D. Geman. 1984. Stochastic Relaxation, Gibbs Distribution and The Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 6(6): pp. 721–741.
- Gildea, D. 2003. Loosely Tree-based Alignment for Machine Translation. *Proceedings of the ACL companion volume*, pp. 80–87. Sapporo, Japan.
- Gildea, D. and D. Jurafsky. 2002. Automatic Labeling of Semantic Roles. *Computational Linguistics* 28(3): pp. 256--288.
- Gimenez, J. and L. Marquez. 2008. A Smorgasbord of Features for Automatic MT Evaluation. *3rd ACL Workshop on Statistical Machine Translation*, pp. 195-198. Columbus, OH.
- Gimenez, J. and L. Marquez. 2007a. Context-aware Discriminative Phrase Selection for Statistical Machine Translation. *Proceedings of the Workshop on Statistical Machine Translation*. Prague.
- Gimenez, J. and L. Marquez. 2007b. Linguistic Features for Automatic Evaluation of Heterogenous MT Systems. *Proceedings of the Second Workshop on Statistical Machine Translation*, pp. 256-264. Prague, Czech Republic.
- Goldwater, S., T. L. Griffiths and M. Johnson. 2006. Contextual Dependencies. Unsupervised Word Segmentation. *Proceedings of ACL/Coling*. Sydney, Australia.
- Goldwater, S. and D. McClosky. 2005. Improving Statistical MT through Morphological Analysis. *Proceedings of HLT/EMNLP*, pp. 676–683. Vancouver, Canada.
- Good, I.J. 1953. The Population Frequencies of Species and the Estimation of Population Parameters. *Biometrika* 40(3 and 4): pp. 237–264.
- Goodman, J.T. 2001. A Bit of Progress in Language Modeling (extended version). *Technical Report MSRTR-2001-72, Machine Learning and Applied Statistics Group*, Microsoft Research: Redmond, WA.
- Goodman, J. 1999. Semiring Parsing. *Computational Linguistics* 25: pp. 573–605.
- Graehl, J. and K. Knight. 2004. Training Tree Transducers. *Proceedings of HLT/NAACL*.
- Gropp, W., E. Lusk and A. Skjellum. 1999. *Using MPI: Portable Parallel Programming with the Message-passing Interface*. MIT Press: Cambridge, MA.
- Gupta, D. and N. Chatterjee. 2003. Identification of Divergence for English to Hindi EBMT. *Proceedings of the MT Summit IX*.
- Habash, N., A. Soudi and T. Buckwalter. 2007. On Arabic Transliteration. A. van den Bosch and A. Soudi, eds., *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Springer.
- Habash, N. 2007a. Arabic Morphological Representations for Machine Translation. A. van den Bosch and A. Soudi, eds., *Arabic Computational Morphology: Knowledge-based and Empirical Methods*. Springer.
- Habash, N. 2007b. Syntactic Preprocessing for Statistical Machine Translation. *Proceedings of the MT Summit XI*.

- Habash, N. and F. Sadat. 2006. Arabic Preprocessing Schemes for Statistical Machine Translation. *Proceedings of HLT-NAACL*, pp. 49–52. New York, NY.
- Habash, N. and O. Rambow. 2005. Arabic Tokenization, Part-of-Speech Tagging and Morphological Disambiguation in One Fell Swoop. *Proceedings of ACL*, pp. 573–580. Ann Arbor, MI.
- Habash, N. 2004. Large Scale Lexeme Based Arabic Morphological Generation. *Proceedings of Conference sur le Traitement Automatique du Langage Naturel*, pp. 271–276. Fez, Morocco.
- Hakkani-Tur, D., G. Riccardi and A. Gorin. 2002. Active Learning for Automatic Speech Recognition. *Proceedings of IEEE - ICASSP*.
- Hasan, S., R. Zens and H. Ney. 2007. Are Very Large N-best Lists Useful for SMT? *Proceedings of HLT/NAACL Companion Volume, Short Papers*, pp. 57–60. Rochester, NY.
- Hassan, H., K. Sima'an and A. Way. 2007. Supertagged Phrase-based Statistical Machine Translation. *Proceedings of ACL*, pp. 288–295. Prague, Czech Republic.
- He, X., M. Yang, J. Gao, P. Nguyen and R. Moore. 2008. Indirect-HMM-based Hypothesis Alignment for Combining Outputs from Machine Translation Systems. *Proceedings of EMNLP*, pp. 98–107.
- Held and Karp. 1962. A Dynamic Programming Approach to Sequencing Problems. *SIAM* 10(1): pp. 196–210.
- Hewavitharana, S., A. Lavie and S. Vogel. 2007. Experiments with a Noun-Phrase Driven Statistical Machine Translation System. *Proceedings of the MT Summit XI*.
- Hildebrand, A.S. and S. Vogel. 2008. Combination of Machine Translation Systems via Hypothesis Selection from Combined N-best Lists. *Proceedings of AMTA*, pp. 254–261. Waikiki, Hawaii.
- Hildebrand, A.S., M. Eck, S. Vogel and A. Waibel. 2005. Adaptation of the Translation Model for Statistical Machine Translation Based on Information Retrieval. *Proceedings of EAMT*, pp. 133–142, Budapest, Hungary.
- Hoang, H. and P. Koehn. 2009. Improving Mod-range Reordering using Templates of Factors. *Proceedings of EACL*. Athens, Greece.
- Huang, C. R. and K. J. Chen. 1995. Academic Sinica Balanced Corpus. *Technical Report 95-02/98-04, Academic Sinica*. Taipei, Taiwan.
- Huang, F. 2009. Confidence Measure For Word Alignment. *Proceedings of ACL/AFNLP*, pp. 932–940.
- Huang, L. 2008. Forest Reranking: Discriminative Parsing with Non-local Features. *Proceedings of ACL*.
- Huang, L. and D. Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. *Proceedings of ACL* pp. 144–151. Prague, Czech Republic.
- Huang, L. and D. Chiang. 2005. Better K-best Parsing. *Proceedings of the 9th International Workshop on Parsing Technologies*.

- Huang, L., K. Knight and A. Joshi. 2006. Statistical Syntax-directed Translation with Extended Domain of Locality. *Proceedings of AMTA*. Boston, MA.
- Huang, Z., D. Filimonov and M. Harper. 2008. Accuracy Enhancements for Mandarin Parsing. *Technical report, University of Maryland*.
- Hwa, R. 2004. Sample Selection for Statistical Parsing. *Computational Linguistics* 30(3).
- Ittycheriah, A. and S. Roukos. 2005. A Maximum Entropy Word Aligner for Arabic-English Machine Translation. *Proceedings of HLT/EMNLP*, pp. 89–96.
- Ittycheriah, A. and S. Roukos. 2007. Direct Translation Model 2. *Proceedings of HLT/NAACL*, pp. 57–64. Rochester, New York.
- IWSLT. 2007. International Workshop on Spoken Language Translation Home Page. <http://www.slt.atr.jp/IWSLT2007>.
- Iyer, R. and M. Ostendorf. 1999. Relevance Weighting for Combining Multi-domain Data for *N*-gram Language Modeling. *Computer Speech and Language* 13(3): pp. 267–282.
- Jayaraman, S. and A. Lavie. 2005. Multi-Engine Machine Translation Guided by Explicit Word Matching. *Proceedings of EAMT*, pp.143-152.
- Jelinek, F. 1998. *Statistical Methods for Speech Recognition*. MIT Press: Cambridge, MA.
- Jiang, W., J. Zhao, Y. Guan and Z. Xu. 2005. Chinese Word Segmentation Based on Mixing Model. *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.
- Jurafsky, D. and J.H. Martin. 2000. *Speech and Language Processing*. Prentice Hall.
- Kamm, T.M. and G.G.L. Meyer. 2002. Selective Sampling of Training Data for Speech Recognition. *Proceedings of HLT*.
- Karakos, D., J. Eisner, S. Khudanpur and M. Dreyer. 2008. Machine Translation System Combination using ITG-based Alignments. *Proceedings of ACL/HLT*.
- Kashioka, H., T. Maruyama and H. Tanaka. 2003. Building a Parallel Corpus for Monologue with Clause Alignment. *Proceedings of the MT Summit IX*.
- Kilgarriff, A. 2001. English Lexical Sample Task Description. *Proceedings of Senseval-2, The Second International Workshop on Evaluating Word Sense Disambiguation Systems*, pp. 17-20. Toulouse, France.
- Kilgarriff, A. and J. Rosenzweig. 1999. Framework and Results for English Senseval. *Computers and the Humanities* 34(1): pp. 15-48. Special issue on SENSEVAL.
- Kirchhoff, K. and M. Yang. 2005. Improved Language Modeling for Statistical Machine Translation. *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pp. 125–128, Ann Arbor, MI.
- Klein, D. and C.D. Manning. 2003. A* Parsing: Fast Exact Viterbi Parse Selection. *Proceedings of ACL/HLT*, pp.40-47.
- Kneser, R. and H. Ney. 1995. Improved Backing-off for *M*-gram Language Modeling. *Proceedings of ICASSP*, pp. 181–184.

- Knight, K. 1999. Decoding Complexity in Word Replacement Translation Models. *Computational Linguistics* 25(4): pp. 607-615.
- Knight, K. and J. Graehl. 2005. An Overview of Probabilistic Tree Transducers for Natural Language Processing. *Proceedings of CICLing*.
- Koehn, P. 2009. *Statistical Machine Translation*. Cambridge University Press.
- Koehn, P. and H. Hoang. 2007. Factored Translation Models. *Proceedings of EMNLP-CoNLL*, pp. 868-876.
- Koehn, P., H. Hoang, A. Birch, C. Callison-Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin and E. Herbst. 2007. Moses: Open Source Toolkit For Statistical Machine Translation. *Proceedings of ACL* pp.177-180, Prague, Czech Republic.
- Koehn, P., A. Axelrod, A.B. Mayne, C. Callison-Burch, M. Osborne and D. Talbot. 2005. Edinburgh System Description for the 2005 IWSLT Speech Translation Evaluation. *Proceedings of IWSLT*.
- Koehn, P. 2004a. Pharaoh: a Beam Search Decoder for Phrase-based Statistical Machine Translation Models. *Proceedings of AMTA*, pp. 115-124. Washington, DC
- Koehn, P. 2004b. Statistical Significance Tests for Machine Translation Evaluation. *Proceedings of EMNLP*. Barcelona, Spain.
- Koehn, P. and K. Knight. 2003. Feature-rich Statistical Translation of Noun Phrases. E. Hinrichs and D. Roth, eds. *Proceedings of ACL*, pp. 311-318.
- Koehn, P., F.J. Och and D. Marcu. 2003. Statistical Phrase-based Translation. *Proceedings of HLT-NAACL*, pp. 127-133, Edmonton, Canada.
- Koehn, P. and K. Knight. 2002. ChunkMT: Machine Translation with Richer Linguistic Knowledge. *Unpublished*.
- Komachi, M., M. Nagata and Y. Matsumoto. 2006. Phrase Reordering for Statistical Machine Translation Based on Predicate-argument Structure. *Proceedings of IWSLT*. Kyoto, Japan.
- Kumar, S. and W. Byrne. 2004. Minimum Bayes-risk Decoding for Statistical Machine Translation. *Proceedings of HLT/NAACL*, pp.169-176.
- Langlais, P. and A. Patry. 2007. Translating Unknown Words by Analogical Learning. *Proceedings of EMNLP-CoNLL*, pp. 877-886.
- LanguageWeaver. 2005. Language Weaver Inc.'s Chinese-to-English MT System, release v4.0. <http://www.languageweaver.com>.
- Lavie, A. and A. Agarwal. 2007. METEOR: An Automatic Metric for MT Evaluation with High Levels of Correlation with Human Judgments. *Proceedings of the Second Workshop on Statistical Machine Translation*, pp.228-231.
- LDC. 2003. Linguistic Data Consortium Chinese Resource Home Page. <http://www.ldc.upenn.edu/Projects/Chinese>.

- Lee, J., D. Lee and G.G. Lee. 2008. Transformation-based Sentence Splitting Method for Statistical Machine Translation. *Proceedings of the Workshop on Technologies and Corpora for Asia-Pacific Speech Translation*.
- Lee, Y.S. 2004. Morphological Analysis for Statistical Machine Translation. *Proceedings of HLT-NAACL*, pp. 57–60. Boston, MA.
- Lee, Y.S., K. Papineni and S. Roukos. 2003. Language Model Based Arabic Word Segmentation. *Proceedings of ACL*, pp. 399–406. Sapporo, Japan.
- Leusch, G., N. Ueffing and H. Ney. 2003. A Novel String-to-String Distance Measure with Applications to Machine Translation Evaluation. *Proceedings of MT Summit IX*, pp. 240-247.
- Levow, G. A. 2006. The Third International Chinese Language Processing Bakeoff: Word Segmentation and Named Entity Recognition. *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*.
- Levow, G.A., D. Oard and P. Resnik. 2005. Dictionary-based Techniques for Cross-Language Information Retrieval. *Information Processing and Management* 41(3): pp. 523–547.
- Li, C. H., M. Li, D. Zhang, M. Li, M. Zhou and Y. Guan. 2007. A Probabilistic Approach To Syntax-Based Reordering for Statistical Machine Translation. *Proceedings of ACL*, pp. 720–727. Prague, Czech Republic.
- Li, Z. and J. Eisner. 2009. First- and Second-order Expectation Semirings with Applications to Minimum-Risk Training on Translation Forests. *Proceedings of EMNLP*.
- Li, Z. and S. Khudanpur. 2009. Efficient Extraction of Oracle-best Translations from Hypergraphs. *Proceedings of NAACL*.
- Li, Z., C. Callison-Burch, C. Dyer, J. Ganitkevitch, S. Khudanpur, L. Schwartz, W. Thornton, J. Weese and O. Zaidan. 2009a. Joshua: An Open Source Toolkit for Parsing-based Machine Translation. *Proceedings of the Fourth Workshop on Statistical Machine Translation*.
- Li, Z., J. Eisner and S. Khudanpur. 2009b. Variational Decoding for Statistical Machine Translation. *Proceedings of ACL*.
- Li, Z. and S. Khudanpur. 2008. Large-scale Discriminative n-gram Language Models for Statistical Machine Translation. *Proceedings of AMTA*.
- Liang, P., A. Bouchard-Cote, D. Klein and B. Taskar. 2006a. An End-to-end Discriminative Approach to Machine Translation. *Proceedings of Coling/ACL*.
- Liang, P., B. Taskar and D. Klein. 2006b. Alignment by Agreement. *Proceedings of HLT/NAACL*, pp.104-111, New York, NY.
- Lopez, A. 2007. Hierarchical Phrase-based Translation with Suffix Arrays. *Proceedings of EMNLP/CoNLL*.
- Lowerre, B. and R. Reddy. 1980. The Harpy Speech Understanding System. *Trends in Speech Recognition*, W.A. Lea, ed. Prentice Hall: Englewood Cliffs, NJ.

- Lu, Y., J. Huang and Q. Liu. 2007. Improving Statistical Machine Translation Performance By Training Data Selection And Optimization. *Proceedings of EMNLP-CoNLL*, pp. 343-350.
- Maamouri, M. and A. Bies. 2004. Developing an Arabic Treebank: Methods, Guidelines Procedures and Tools. *Proceeding of Coling*.
- Magerman, D. 1995. Statistical Decision-tree Models for Parsing. *Proceedings of ACL*.
- Manber, U. and G. Myers. 1993. Suffix Arrays: a New Method for On-Line String Searches. *SIAM J. Comput.* 22(5), pp. 935-948.
- Manber, U. and G. Myers. 1990. Suffix Arrays: a New Method for On-line String Searches. *SODA '90: Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms*, pp. 319-327. Society for Industrial and Applied Mathematics.
- Mangu, L., E. Brill and A. Stolcke. 2000. Finding Consensus in Speech Recognition: Word Error Minimization and Other Applications of Confusion Networks. *Computer Speech and Language* 14(4): pp. 373-400.
- Marcu, D., W. Wang, A. Echihabi and K. Knight. 2006. SPMT: Statistical Machine Translation with Syntactified Target Language Phrases. *Proceedings of EMNLP*, pp. 44-52. Sydney, Australia.
- Marcus, M.P., B. Santorini and M.A. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics* 19(2):313-330.
- Marton, Y. and P. Resnik. 2008. Soft Syntactic Constraints for Hierarchical Phrased-Based Translation. *Proceedings of ACL/HLT*, pp. 1003-1011.
- Matusov, E., G. Leusch and H. Ney. 2009. Learning to Combine Machine Translation Systems. *Learning machine translation (NIP Series)*, pp. 257-276. MIT Press.
- Matusov, E., G. Leusch, R.E. Banchs, N. Bertoldi, D. Dechelotte, M. Federico, M. Kolss, Y.S. Lee, J.B. Marino, M. Paulik, S. Roukos, H. Schwenk and H. Ney. 2008. System Combination for Machine Translation of Spoken and Written Language. *IEEE Transactions on Audio, Speech and Language Processing* 16(7): pp. 1222-1237.
- Matusov, E., N. Ueffing and H. Ney. 2006. Computing Consensus Translation from Multiple Machine Translation Systems Using Enhanced Hypotheses Alignment. *Proceedings of EACL*, pp. 33-40.
- Matusov, E., R. Zens and H. Ney. 2004. Symmetric Word Alignments for Statistical Machine Translation. *Proceedings of Coling*, pp. 219-225.
- McDonald, R., K. Crammer and F. Pereira. 2005. Online Large-margin Training of Dependency Parsers. *Proceedings of ACL*.
- Melamed, I.D. 2004. Statistical Machine Translation by Parsing. *Proceedings of ACL*, pp. 653-660, Barcelona, Spain.
- Melamed, I.D., G. Satta and B. Wellington. 2004. Generalized Multitext Grammars. *Proceedings of ACL*. Barcelona, Spain.
- Mellebeek, B., K. Owczarzak, D. Groves, J. van Genabith and A. Way. 2006. A Syntactic Skeleton For Statistical Machine Translation. *Proceedings of EAMT*. Oslo, Norway.

- Mihalcea, R., T. Chklovski and A. Killgariff. 2004. The Senseval-3 English Lexical Sample Task. *Proceedings of Senseval-3*. pp. 25-28. Barcelona, Spain.
- Miller, G. 1990. Wordnet: An On-line Lexical Database. *International Journal of Lexicography* 3(4): pp. 235–244.
- Minkov, E., K. Toutanova and H. Suzuki. 2007. Generating Complex Morphology For Machine Translation. *Proceedings of ACL*, pp. 128–135. Prague, Czech Republic.
- Minkov, E., R. Wang, A. Tomasic and W. Cohen. 2006. NER Systems that Suit Users' Preferences: Adjusting the Recall-Precision Trade-Off for Entity Extraction. *Proceedings of NAACL-HLT*, Companion Volume: Short Papers, New York, NY..
- Moore, R.C. 2005. A Discriminative Framework for Bilingual Word Alignment. *Proceedings of HLT/EMNLP*, pp. 81-88. Morristown, NJ.
- Moore, R.C. and C. Quirk. 2007. Faster Beam-search Decoding for Phrasal Statistical Machine Translation. *Proceedings of the Machine Translation Summit XI*. Copenhagen, Denmark.
- Munteanu, D.S. and D. Marcu. 2005. Improving Machine Translation Performance By Exploiting Comparable Corpora. *Computational Linguistics* 31(4): pp. 477-504.
- Murata, M., K. Uchimoto, Q. Ma and H. Isahara. 2001. Using a Support-vector Machine For Japanese-to-English Translation of Tense, Aspect and Modality. *Proceedings of ACL*.
- Ney, H., D. Mergel, A. Noll and A. Paeseler. 1992. Data Driven Search Organization for Continuous Speech Recognition in the SPICOS System. *IEEE Transaction on Signal Processing* 40(2): pp. 272–281.
- Ng, H.T. and J.K. Low. 2004. Chinese Part-of-speech Tagging: One-at-a-time or All-at-once? Word-Based or Character-based? *Proceedings of EMNLP*.
- Nguyen, T. P. and A. Shimazu. 2006. Improving Phrase-based Statistical Machine Translation with Morpho-syntactic Analysis and Transformation. *Proceedings of AMTA*. Boston, MA.
- Niehues, J. and S. Vogel. 2008. Discriminative Word Alignment via Alignment Matrix Modeling. *Proceedings of the Third Workshop on Statistical Machine Translation*, pp. 18–25. Columbus, OH.
- Nießen, S. and H. Ney. 2004. Statistical Machine Translation with Scarce Resources using Morpho-syntactic Information. *Computational Linguistics* 30(2).
- Nießen, S. and H. Ney. 2001. Toward Hierarchical Models For Statistical Machine Translation Of Inflected Languages. *Proceedings of ACL*, pp. 47–54.
- Nikoulina, V. and M. Dymetman. 2008. Experiments in Discriminating Phrase-based Translations on the Basis of Syntactic Coupling Features. *Proceedings of ACL/HLT, the Second Workshop on Syntax and Structure in Statistical Translation*, pp. 55–60. Columbus, OH.
- Och, F. J., D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin and D. Radev. 2004. A Smorgasbord of Features for Statistical Machine Translation. *Proceedings of HLT-NAACL*.

- Och, F.J. and H. Ney. 2004. The Alignment Template Approach to Statistical Machine Translation. *Computational Linguistics* 30(4): pp. 417–450.
- Och, F.J. 2003. Minimum Error Rate Training in Statistical Machine Translation. *Proceedings of ACL*, pp. 160–167. Sapporo, Japan.
- Och, F. J. and H. Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics* 29(1): pp. 19–51.
- Och, F. J. and Ney, H. 2002. Discriminative Training and Maximum Entropy Models for Statistical Machine Translations. *Proceedings of ACL*, pp. 295–302. Philadelphia, PA.
- Och, F. J. and H. Ney. 2000. Improved Statistical Alignment Models. *Proceedings of ACL*.
- Och, F.J., C. Tillmann and H. Ney. 1999. Improved Alignment Models for Statistical Machine Translation. *Proceedings of the Joint SIGDAT Conf. on Empirical Methods in Natural Language Processing and Very Large Corpora*, pp. 20-28. College Park, MD.
- Odell, J. 1995. *The Use of Context in Large Vocabulary Speech Recognition*. Ph.D. Thesis, Cambridge University Engineering University.
- Oflazer, K. and I. Durgar El-Kahlout. 2007. Exploring Different Representational Units in English-to-Turkish Statistical Machine Translation. *Proceedings of the Second Workshop on Statistical Machine Translation*, pp. 25–32. Prague, Czech Republic.
- Olive, J. 2005. *Global Autonomous Language Exploitation (GALE)*. DARPA/IPTO Proposer Information Pamphlet.
- Ostendorf, M., A. Kannan, S. Austin, O. Kimball, R. Schwartz and J.R. Rohlicek. 1991. Integration of Diverse Recognition Methodologies through Reevaluation of N-best Sentence Hypotheses. *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*, pp. 83-87.
- Owczarzak, K., B. Mellebeek, D. Groves, J. van Genabith and A. Way. 2006. Wrapper Syntax For Example-based Machine Translation. *Proceedings of AMT*.
- Palmer, M., D. Gildea and P. Kingsbury. 2005. The Proposition Bank: an Annotated Corpus of Semantic Roles. *Computational Linguistics* 31(1): pp. 71-105
- Papineni, K., S. Roukos, T. Ward and W.J. Zhu. 2002. Bleu: a Method For Automatic Evaluation Of Machine Translation. *Proceedings of ACL*, pp. 311-318, Philadelphia, PA.
- Papineni, K., S. Roukos and T. Ward. 2001. Bleu: a method for automatic evaluation of machine translation. *IBM Research Report, RC22176*.
- Papineni, K., S. Roukos and R.T. Ward. 1997. Feature-based Language Understanding. *Proceedings of EUROSPEECH*, pp. 1435–1438. Rhodes, Greece.
- Paul, M., E. Sumita and S. Yamamoto. 2002. Corpus-based Generation of Numeral Classifiers Using Phrase Alignment. *Proceedings of Coling*.
- Paul, Michael. 2006. Overview of the IWSLT06 Evaluation Campaign. *Proceedings of IWSLT*. Kyoto, Japan.

- Paulik, M., K. Rottmann, J. Niehues, S. Hildebrand and S. Vogel. 2007. The ISL Phrase-based MT System for the 2007 ACL Workshop on SMT. *Proceedings of ACL Second Workshop on SMT*.
- Pearl, J. 1988. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, Publishers.
- Ponte, J.M. and W.B. Croft. 1998. A Language Modeling Approach to Information Retrieval. *Research and Development in Information Retrieval*, pp. 275–281.
- Popovic, M., D. Vilar, H. Ney, S. Jovicic and Z. Saric. 2005. Augmenting a Small Parallel Text with Morpho-syntactic Language. *Proceedings of the ACL Workshop on Building and Using Parallel Texts*, pp. 41–48. Ann Arbor, MI.
- Pradhan, S., K. Hacioglu, V. Krugler, W. Ward, J.H. Martin and D. Jurafsky. 2005. Support Vector Learning for Semantic Argument Classification. *Machine Learning* 60(1-3): pp. 11--39.
- Press, W.H., S.A. Teukolsky, W.T. Vetterling and B.P. Flannery. 2007. Numerical Recipes: the Art Of Scientific Computing. *Cambridge University Press*. 3rd Edition.
- Press, W. H., S. A. Teukolsky, W. T. Vetterling and B. P. Flannery. 2002. Numerical Recipes in C++. *Cambridge University Press*: Cambridge, UK.
- Pritchard, J., M. Stephens and P. Donnell. 2000. Inference of Population Structure Using Multilocus Genotype Data. *Genetics* 155: pp. 945–959.
- Quirk, C., A. Menezes and C. Cherry. 2005. Dependency Treelet Translation: Syntactically Informed Phrasal SMT. *Proceedings of ACL*, pp. 271–279. Ann Arbor, MI.
- Ramanathan, A., J. Hegde, R.M. Shah, P. Bhattacharyya and M. Sasikumar. 2008. Simple Syntactic And Morphological Processing Can Help English-Hindi Statistical Machine Translation. *Proceedings of the International Joint Conference on Natural Language Processing*.
- Resnik, P. and N. A. Smith. 2003. The Web as a Parallel Corpus. *Computational Linguistics* 29(3): pp. 349-380.
- Riesa, J. and D. Yarowsky. 2006. Minimally Supervised Morphological Segmentation with Applications to Machine Translation. *Proceedings of AMTA*. Boston, MA.
- Roark, B., M. Saraclar and M. Collins. 2007. Discriminative N-gram Language Modeling. *Computer Speech and Language* 21(2): pp. 373-392.
- Rosenfeld, R. 1994. Adaptive Statistical Language Modeling: A Maximum Entropy Approach. *Ph.D. thesis, Computer Science Department, Carnegie Mellon University*, Pittsburgh, PA, April. TR CMU-CS-94-138.
- Rosti, A.V.I., B. Zhang, S. Matsoukas and R. Schwartz. 2009. Incremental Hypothesis Alignment with Flexible Matching for Building Confusion Networks: BBN System Description for WMT09 System Combination Task. *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pp.61-65.
- Rosti, A.V.I, B. Zhang, S. Matsoukas and R. Schwartz. 2008. Incremental Hypothesis Alignment for Building Confusion Networks with Application to Machine Translation

- System Combination. *Proceedings of the Third Workshop on Statistical Machine Translation*, pp.183—186.
- Rosti, A.V.I., S. Matsoukas and R. Schwartz. 2007a. Improved Word-Level System Combination for Machine Translation. *Proceedings of ACL*, pp.312-319.
- Rosti, A.V.I., B. Xiang, S. Matsoukas, R. Schwartz, N.F. Ayan and B.J. Dorr. 2007b. Combining Outputs from Multiple Machine Translation Systems. *Proceedings of HLT/NAACL*, pp.228-235.
- Sadat F. and N. Habash. 2006. Combination of Arabic Preprocessing Schemes for Statistical Machine Translation. *Proceedings of Coling/ACL*, pp. 1–8, Sydney, Australia.
- Sadat, F., H. Johnson, A. Agbago, G. Foster, R. Kuhn, J. Martin and A. Tikuisis. 2005. Portage: A Phrase-based Machine Translation System. *Proceedings of the ACL 2005 Workshop on Building and Using Parallel Texts*. Ann Arbor, MI..
- Saiz, J.C. 2008. Novel Statistical Approaches to Text Classification, Machine Translation and Computer-Assisted Translation. *Ph.D. thesis, Universidad Politecnica de valencia*.
- Saon, G., D. Povey and G. Zweig. 2005. Anatomy of an Extremely Fast LVCSR Decoder. *Proceedings of Interspeech*.
- Schafer, C. and D. Yarowsky. 2003a. Statistical Machine Translation Using Coercive Two-Level Syntactic Transduction. *Proceedings of EMNLP*, pp. 9–16.
- Schafer, C. and D. Yarowsky. 2003b. A Two-level Syntax-Based Approach to Arabic-English Statistical Machine Translation. *Proceedings of the MT Summit IX Workshop on Machine Translation for Semitic Languages: Issues and Approaches*.
- Schiehlen, M. 1998. Learning Tense Translation From Bilingual Corpora. *Proceedings of ACL*.
- Schohn, G. and D. Cohn. 2000. Less Is More: Active Learning with Support Vector Machines. *Proceedings of 17th International Conference on Machine Learning*, pp. 839-846.
- Seung, H.S., M. Opper and H. Sompolinsky. 1992. Query by Committee. *Computational Learning Theory*, pp. 287-294.
- Shapira, D. and J.A. Storer. 2002. Edit Distance with Move Operations. *Proceedings of the 13th Annual Symposium on Combinatorial Pattern Matching*, pp. 85-98.
- Shen, L., J. Xu and R. Weischedel. 2008. A New String-to-Dependency Machine Translation Algorithm With a Target Dependency Language Model. *Proceedings of ACL*.
- Shen, L., A. Sarkar and F.J. Och. 2004. Discriminative Reranking for Machine Translation. *Proceedings of HLT/NAACL*.
- Shen, W., R. Zens, N. Bertoldi and M. Federico. 2006. The JHU Workshop 2006 IWSLT System. *Proceedings of IWSLT*. Kyoto, Japan.
- Shen, Y., C. Lo, M. Carpuat and D. Wu. 2007. HKUST Statistical Machine Translation Experiments for IWSLT 2007. *Proceedings of IWSLT*, pp. 84-88. Trento, Italy.

- Shi, Y. and M. Wang. 2007. A Dual-layer CRFS Based Joint Decoding Method For Cascaded Segmentation and Labeling Tasks. *Proceedings of the International Joint Conference on Artificial Intelligence*.
- Shieber, S., Y. Schabes and F. C. N. Pereira. 1995. Principles and Implementation of Deductive Parsing. *Journal of Logic Programming* 24: pp. 3–36.
- Shieber, S. and Y. Schabes. 1990. Synchronous Tree Adjoining Grammars. *Proceedings of COLING The 13th Int. Confermce on Computational Linguistics*.
- Sim, K.C., W.J. Byrne, M.J.F. Gales, H. Sahbi and P.C. Woodland. 2007. Consensus Network Decoding for Statistical Machine Translation System Combination. *Proceedings of IEEE ICASSP*, pp.105-108.
- Simard, M., C. Goutte and P. Isabelle. Statistical Phrase-based Post-Editing. 2007a. *Proceedings of HLT-NAACL*, pp. 508-515.
- Simard, M., N. Ueffing, P. Isabelle and R. Kuhn. 2007b. Rule-based Translation with Statistical Phrase-based Postediting. *Proceedings of the 2nd ACL Workshop on Statistical Machine Translation*, pp. 203-206.
- Sjolander, K., K. Karplus, M. Brown, R. Hughey, A. Krogh, I.S. Mian and D. Haussler. 1996. Dirichlet Mixtures: A Method for Improving Detection of Weak but Significant Protein Sequence Homology. *Computer Applications in the Bio-sciences* 12.
- Smith, D.A. and J. Eisner. 2006. Minimum Risk Annealing for Training Log-linear Models. *Proceedings of ACL*.
- Smith, N.A. and J. Eisner. 2004. Annealing Techniques for Unsupervised Statistical Language Learning. *Proceedings of ACL*.
- Snover, M., N. Madhani, B. Dorr and R. Schwartz. 2009. Fluency, Adequacy, or HTER? Exploring Different Human Judgments with a Tunable MT Metric. *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pp.259-268.
- Snover, M., B. Dorr, R. Schwartz, L. Micciula and J. Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. *Proceedings of AMTA*, pp. 223—231.
- Snover, M., B. Dorr, R. Schwartz, J. Makhoul, L. Micciula and R. Weischedel. 2005. A Study of Translation Error Rate with Targeted Human Annotation. Technical Report LAMPTR- 126, CS-TR-4755,UMIACS-TR-2005-58, University of Maryland, College Park and BBN Technologies.
- Steinbiss, V., B.H. Tran and H. Ney. 1994. Improvements in Beam Search. *Proceedings of ICSLP*, pp. 2143-2146.
- Stolcke, A. 2002. SRILM - an Extensible Language Modeling Toolkit. *Proceedings of ICSLP*, pp. 901–904. Denver, CO.
- Stolcke, A. 1998. Entropy-based Pruning of Backoff Language Models. *Proceedings of the DARPA Broadcast News Transcription and Understanding Workshop*.
- Sun, H. and D. Jurafsky. Shallow Semantic Parsing of Chinese. *Proceedings of HLT/NAACL*, pp. 249-256, Boston, MA.

- Sutton, C., M. Sindelar and A. McCallum. 2006. Reducing Weight Undertraining in Structured Discriminative Learning. *Proceedings of HLT/NAACL*.
- Suzuki, H. and K. Toutanova. 2006. Learning to Predict Case Markers In Japanese. *Proceedings of Coling/ACL*. Sydney, Australia.
- Talbot, D. and M. Osborne. 2006. Modeling Lexical Redundancy For Machine Translation. *Proceedings of Coling/ACL*, pp. 969–976, Sydney, Australia.
- Tam, Y.C., I. Lane and T. Schultz. 2007. Bilingual-LSA Based LM adaptation for Spoken Language Translation. *Proceedings of ACL*. Prague, Czech Republic.
- Tanaka, T. and T. Baldwin. 2003. Translation Selection for Japanese-English Noun-noun Compounds. *Proceedings of the MT Summit IX*.
- Tantug, A. C., E. Adali and K. Oflazer. 2007a. Machine Translation between Turkic Languages. *Proceedings of ACL companion volume*: pp. 189–192. Prague, Czech Republic.
- Tantug, A. C., E. Adali and K. Oflazer. 2007b. A MT System from Turkmen to Turkish Employing Finite State and Statistical Methods. *Proceedings of the MT Summit XI*.
- Taskar, B., S. Lacoste-Julien and D. Klein. 2005. A Discriminative Matching Approach to Word Alignment. *Proceedings of HLT/EMNLP*, pp. 73–80. Morristown, NJ, USA.
- Tillmann, C. 2008. A Rule-driven Dynamic Programming Decoder for Statistical MT. *Proceedings of Second Workshop on Syntax and Structure in Statistical Translation, at ACL/HLT*, pp. 37–45. Columbus, OH.
- Tillmann, C. 2006. Efficient Dynamic Programming Search Algorithms for Phrase-based SMT. *Proceedings of the Workshop on Computationally Hard Problems and Joint Inference in Speech and Language Processing*, pp. 9-16, New York, NY.
- Tillmann, C. 2004. A Unigram Orientation Model for Statistical Machine Translation. *Proceedings of HLT/NAACL*.
- Tillmann, C. and H. Ney. 2003. Word Reordering and a DP Beam Search Algorithm for Statistical Machine Translation. *Computational Linguistics* 29(1):97–133.
- Tillmann, C. 2003. A Projection Extension Algorithm for Statistical Machine Translation. *Proceedings of EMNLP*, pp. 1–8. Sapporo, Japan.
- Tillmann, C. 2001. *Word Re-Ordering and Dynamic Programming based Search Algorithm for Statistical Machine Translation*. Ph.D. thesis. University of Technology, Aachen, Germany.
- Toutanova, K., H. Suzuki and A. Ruopp. 2008. Applying Morphology Generation Models to Machine Translation. *Proceedings of ACL/HLT*, pp. 514–522. Columbus, OH.
- Toutanova, K. and H. Suzuki. 2007. Generating Case Markers in Machine Translation. *Proceedings of HLT/NAACL*, pp. 49–56, Rochester, NY.
- Tromble, R.W. and S. Kumar and F.J. Och and W. Macherey. 2008. Lattice Minimum Bayes-Risk Decoding for Statistical Machine Translation. *Proceedings of EMNLP*.

- Tseng, H., P. Chang, G. Andrew, D. Jurafsky and C. Manning. 2005. A Conditional Random Field Word Segmenter. *Proceedings of the Fourth SIGHAN Workshop on Chinese Language Processing*.
- Tur, G., D. Hakkani-Tur and R.E. Schapire. 2005. Combining Active and Semi-supervised Learning for Spoken Language Understanding. *Journal of Speech Communication* 45(2): pp. 171-186.
- Ueffing, N., J. Stephan, E. Matusov, L. Dugast, G. Foster, R. Kuhn, J. Senellart and J. Yang. 2008. Tighter Integration of Rule-based and Statistical MT in Serial System Combination. *Proceedings of Coling*. Manchester, U.K.
- Ueffing, N. and H. Ney. 2007. Word-level Confidence Estimation for Machine Translation. *Computational Linguistics* 33(1): pp. 9–40.
- Ueffing, N., M. Simard, S. Larkin and J. H. Johnson. 2007. NRC's PORTAGE System for WMT 2007. *Proceedings of 2nd ACL Workshop on Statistical Machine Translation*, pp. 185-188.
- Ueffing, N. and H. Ney. 2003. Using POS Information for SMT into Morphologically Rich Languages. *Proceedings of EACL*.
- Vanden Berghen, F. and H. Bersini. 2005. CONDOR, A New Parallel, Constrained Extension of Powell's UOBYQA Algorithm: Experimental Results and Comparison with the DFO Algorithm. *Journal of Computational and Applied Mathematics* 181: pp. 157-175.
- Vapnik, V. 2006. *Estimation of Dependencies Based on Empirical Data* (Second Edition). Springer, 2006.
- Vilar, D., D. Stein and H. Ney. 2008. Analysing Soft Syntax Features and Heuristics for Hierarchical Phrase Based Machine Translation. *Proceedings of IWSLT*, pp. 190–197. Waikiki, HI.
- Vilar, D., J.T. Peter and H. Ney. 2007. Can We Translate Letters? *Proceedings of the Second Workshop on Statistical Machine Translation*, pp. 33–39. Prague, Czech Republic.
- Vilar, D., M. Popovic and H. Ney. 2006. AER: Do We Need to “Improve” Our Alignments? *Proceedings of IWSLT*, pp. 205–212. Kyoto, Japan.
- Virpioja, S., J.J. Vrynen, M. Creutz and M. Sadeniemi. 2007. Morphology-aware Statistical Machine Translation Based on Morphs Induced in an Unsupervised Manner. *Proceedings of the MT Summit XI*.
- Vogel, S., H. Ney and C. Tillmann. 1996. HMM-based Word Alignment in Statistical Machine Translation. *Proceedings of Coling*, pp. 836–841. Copenhagen, Denmark.
- Wang, C., M. Collins and P. Koehn. 2007. Chinese Syntactic Reordering for Statistical Machine Translation. *Proceedings of EMNLP-CoNLL*, pp. 737–745.
- Wang, W., K. Knight and D. Marcu. 2007. Binarizing Syntax Trees to Improve Syntax-based Machine Translation Accuracy. *Proceedings of EMNLP-CoNLL*.

- Watanabe, T., J. Suzuki, H. Tsukada and H. Isozaki. 2007. Online Large-margin Training for Statistical Machine Translation. *Proceedings of EMNLP-CoNLL*.
- Woodland, P.C. and D. Povey. 2002. Large Scale Discriminative Training of Hidden Markov Models for Speech Recognition. *Computer Speech and Language* 16(1): pp. 25-47.
- Wu, D. and D. Chiang, editors. 2009. *Proceedings of SSST-3, Third Workshop on Syntax and Structure in Statistical Translation, at NAACL-HLT2009*.
- Wu, D. and D. Chiang, editors. 2007. *Proceedings of SSST, NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*.
- Wu, D. and H. Wong. 1998. Machine Translation with a Stochastic Grammatical Channel. *Proceedings of ACL/Coling*, pp. 1408–1415. Montreal, Canada.
- Wu, D. 1997. Stochastic Inversion Transduction Grammars and Bilingual Parsing of Parallel Corpora. *Computational Linguistics* 23(3): pp. 377-403.
- Xia, F. and M. McCord. 2004. Improving a Statistical MT System with Automatically Learned Rewrite Patterns. *Proceedings of Coling*, pp. 508–514. Geneva, Switzerland.
- Xing, E.P., M.I. Jordan and S. Russell. 2003. A Generalized Mean Field Algorithm for Variational Inference in Exponential Families. Meek and Kjaelff, editors, *Uncertainty in Artificial Intelligence (UAI2003)*, pp. 583–591. Morgan Kaufmann Publishers.
- Xu, J., J. Gao, K. Toutanova and H. Ney. 2008. Bayesian Semi-supervised Chinese Word Segmentation for Statistical Machine Translation. *Proceedings of Coling*. Manchester, UK.
- Xu, J., E. Matusov, R. Zens and H. Ney. 2005a. Integrated Chinese Word Segmentation in Statistical Machine translation. *Proceedings of IWSLT*.
- Xu, J., R. Zens and H. Ney. 2005b. Sentence Segmentation using IBM Word Alignment Model 1. *Proceedings of EAMT*, pp. 280–287. Budapest, Hungary.
- Xu, J., R. Zens and H. Ney. 2004. Do We Need Chinese Word Segmentation for Statistical Machine Translation? *Proceedings of the SIGHAN Workshop on Chinese Language Learning*, pp. 122–128. Barcelona, Spain.
- Xue, N. and M. Palmer. 2005. Automatic Semantic Role Labeling for Chinese Verbs. *Proceedings of International Joint Conference on Artificial Intelligence*, Edinburgh, 2005.
- Xue, N. Semantic Role Labeling of Nominalized Predicates in Chinese. 2006. *Proceedings of HLT-NAACL*, pp.431-438. New York, NY.
- Yamada, K. and K. Knight. 2002. A Decoder for Syntax-based Statistical MT. *Proceedings of ACL*, pp. 303–310. Philadelphia, PA.
- Yamada, K. and K. Knight. 2001. A Syntax-based Statistical Translation Model. *Proceedings of ACL*.
- Yamamoto, M. and K.W. Church. 2001. Using Suffix Arrays to Compute Term Frequency and Document Frequency for All Substrings in a Corpus. *Computational Linguistics* 27(1): pp. 1–30.

- Yang, J., J. Senellart and R. Zajac. 2003. SYSTRAN's Chinese Word Segmentation. *Proceedings of Second SIGHAN Workshop on Chinese Language Processing*.
- Yang, M. and K. Kirchoff. 2006. Phrase-based Backoff Models for Machine Translation of Highly Inflected Languages. *Proceedings of ACL*. Trento, Italy.
- Ye, Y., K.M. Schneider and S. Abney. 2007. Aspect Marker Generation in English-to-Chinese Machine Translation. *Proceedings of the MT Summit XI*.
- Ye, Y., V.L. Fossum and S. Abney. 2006. Latent Features in Automatic Tense Translation Between Chinese and English. *Proceedings of the Fifth SIGHAN Workshop on Chinese Language Processing*, pp. 48–55, Sydney, Australia.
- Yedidia, J.S., W.T. Freeman and Y. Weiss. 2003. Understanding Belief Propagation and its Generalizations. *Exploring Artificial Intelligence in the New Millennium*, Chap. 8, pp. 239–236, 239–269.
- Zens, R. 2008. Phrase-based Statistical Machine Translation: Models, Search, Training. *Ph.D. thesis, RWTH Aachen University: Aachen, Germany*.
- Zens, R. and H. Ney. 2008. Improvements in Dynamic Programming Beam Search for Phrase-based Statistical Machine Translation. *Proceedings of IWSLT* pp. 198–205. Honolulu, HI.
- Zens, R., S. Hasan and H. Ney. 2007. A Systematic Comparison of Training Criteria for Statistical Machine Translation. *Proceedings of EMNLP/CoNLL*.
- Zens, R. and H. Ney. 2006. N-gram Posterior Probabilities for Statistical Machine Translation. *Proceedings on the Workshop on Statistical Machine Translation*, pp. 72–77. New York, NY.
- Zens, R. and H. Ney. 2004. Improvements in Phrase-based Statistical Machine Translation. *Proceedings of HLT-NAACL*, pp. 257–264. Boston, MA.
- Zens, R., F.J. Och and H. Ney. 2002. Phrase-based Statistical Machine Translation. M. Jarke, J. Koehler and G. Lakemeyer, eds., *Proceedings of the 25th German Conf. on Artificial Intelligence (KI2002)*, volume 2479 of *Lecture Notes in Artificial Intelligence (LNAI)*, pp. 18–32. Springer Verlag: Aachen, Germany.
- Zhang, D., M. Li, N. Duan, C.H. Li and M. Zhou. 2008a. Measure Word Generation for English-Chinese SMT Systems. *Proceedings of HLT/ACL*, pp. 89–96. Columbus, OH..
- Zhang, H., H. Yu, D. Xiong and Q. Liu. 2003. HHMM-based Chinese Lexical Analyzer ICTCLAS. *Proceedings of the Second SIGHAN Workshop on Chinese Language Learning*, pp. 184–187.
- Zhang, H., L. Huang, D. Gildea and K. Knight. 2006. Synchronous Binarization for Machine Translation. *Proceedings of HLT/NAACL*, pp. 256–263. New York, NY.
- Zhang, J., C. Zong and S. Li. 2008b. Sentence Type Based Reordering Model for Statistical Machine Translation. *Proceedings of Coling*, pp. 1089–1096. Manchester, UK.
- Zhang, R., K. Yasuda and E. Sumita. 2008. Chinese Word Segmentation and Statistical Machine Translation. *ACM Transactions on Speech and Language Processing*.

- Zhang, Y., R. Zens and H. Ney. 2007a. Chunk-level Reordering of Source Language Sentences with Automatically Learned Rules for Statistical Machine Translation. *Proceedings of NAACL-HLT 2007 / AMTA Workshop on Syntax and Structure in Statistical Translation*, pp. 1–8, Rochester, NY.
- Zhang, Y., R. Zens and H. Ney. 2007b. Improved Chunk-level Reordering for Statistical Machine Translation. *Proceedings of IWSLT*.
- Zhang, Y., A.S. Hildebrand and S. Vogel. 2006. Distributed Language Modeling for N-best List Re-ranking. *Proceedings of EMNLP*, pp. 216–223. Sydney, Australia.
- Zhang, Y. 2006. Suffix Array and Its Applications in Empirical Natural Language Processing. *Technical Report CMU-LTI-06-010, Language Technologies Institute, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA*.
- Zhang, Y. and S. Vogel. 2005. An Efficient Phrase-to-phrase Alignment Model for Arbitrarily Long Phrase and Large Corpora. *Proceedings of EAMT*. Budapest, Hungary.
- Zhao, B. 2007. Statistical Alignment Models for Translational Equivalences. Ph.D. thesis, Language Technologies Institute, School of Computer Science, Carnegie Mellon University.
- Zhao, B. and E.P. Xing. 2007. HM-BiTAM: Bilingual Topic Exploration, Word Alignment and Translation. *Proceedings of the Neural Information Processing Systems Conference*. Vancouver, Canada.
- Zhao, B. and E.P. Xing. 2006. BiTAM: Bilingual Topic Admixture Models for Word Alignment. *Proceedings of ACL*.
- Zhao, B., N. Ge and K. Papineni. 2005. Innerouter bracket models for word alignment using hidden blocks. *Proceedings of HLT and EMNLP*, pp 177–184.
- Zhao, B., M. Eck and S. Vogel. 2004. Language Model Adaptation for Statistical Machine Translation with Structured Query Models. *Proceedings of Coling*.
- Zhao, T., L. Yajuan, Y. Muyun and Y. Hao. 2001. Increasing Accuracy of Chinese Segmentation with Strategy of Multi-step Processing. *Journal of Chinese Information Processing (Chinese Version)*, 1: pp. 13-18.
- Zheng, J., W. Wang and N.F. Ayan. 2008. Development of SRI's Translation Systems for Broadcast News and Broadcast Conversations. *Proceedings of Interspeech*.
- Zollmann, A. and A. Venugopal. 2006. Syntax Augmented Machine Translation via Chart Parsing. *Proceedings of HLT-NAACL*. New York, NY.
- Zwarts, S. and M. Dras. 2007. Syntax-based Word Reordering in Phrase Based Statistical Machine Translation: Why Does it Work? *Proceedings of the MT Summit XI*.

Chapter 3: Machine Translation from Speech

Editors: Richard Schwartz, Joseph Olive, John McCary, and Caitlin Christianson

3.1 Introduction

Author: Lidia Mangu

This chapter describes approaches for translation from speech. Translation from speech presents two new issues. First, of course, we must recognize the speech in the source language. Although speech recognition has improved considerably over the last three decades, it is still far from being a solved problem. In the best of conditions, when the speech comes from high quality, carefully enunciated speech, on common topics (such as speech read by a trained news broadcaster), the word error rate is typically on the order of 5%. Humans can typically transcribe speech like this with less than 1% disagreement between annotators, so even this best number is still far worse than human performance. However, the task gets much harder when anything changes from this ideal condition. Some of the conditions that cause higher error rate are, if the topic is somewhat unusual, or the speakers are not reading so that their speech is more spontaneous, or if the speakers have an accent or are speaking a dialect, or if there is any acoustic degradation, such as noise or reverberation. In these cases, the word error can increase significantly to 20%, 30%, or higher. Accordingly, most of this chapter discusses techniques for improving speech recognition accuracy, while one section discusses techniques for integrating speech recognition with translation.

The speech recognition process can be divided into three parts: feature computation, acoustic modeling, and language modeling. We compute features related to the short-term power spectrum on a perceptually warped frequency axis. However, in order to enrich the power of the features, we typically concatenate features over several frames. Some methods even look at features over a half second period. 3.2 discusses several methods for feature computation. Several of the sections discuss the use of multi-layer perceptrons to map the long concatenated vector of features from many frames into a manageable vector size. They also discuss methods for combining different features.

Currently, all state of the art speech recognizers use Hidden Markov Models (HMMs) as the basic methodology for performing speech recognition. But there are a large number of ways of using HMMs to model the likely acoustic (feature) sequences that correspond to different phonetic sequences. 3.3 discusses several improvements in the modeling structures and estimation methods that are used within the basic HMM formulation. For example, since the number of parameters needed to model all possible phonetic sequences is potentially extremely large, systems often differ in how they share parameters among different models. There are different estimation criteria that can be

used, from the most basic maximum likelihood techniques that try to estimate a model that maximizes the likelihood of the observed features to models that try to minimize the error rate directly in different ways. In addition, since there are a host of variations on each of these methods, we often see that it is possible to improve recognition accuracy by implementing different recognition systems that use different techniques and then combine their results by different sorts of voting mechanisms. 3.3 discusses some of these techniques.

A key component in the probabilistic formulation for speech recognition is a prior model (a “language model”) that provides a measure of the relative probability for each possible word sequence out of the near infinite number of possibilities. Most typically, we use a Markov chain model (an “ n -gram model”) that estimates the probability of the next word given the previous $n-1$ words. There are variations on this model and on the estimation techniques used to deal with variations in the language. 3.4 discusses some improvements in language modeling techniques.

In general, we try to develop techniques for speech recognition that can be used for any language so that all we need to do to implement a new language is to acquire a source of text data for a language model and a set of transcribed speech in order to estimate the HMM acoustic model. However, there are often some specific details of different languages that must be acknowledged when building a system in order to obtain the lowest possible error rates. For example, Arabic is normally written without the diacritics (short vowels, non-vowels, consonant doubling), which means a typical word-level transcription of speech is missing some phonetic information. In addition, the number of words in Arabic text is very large, because some morphemes that would typically be separate words in English (and, in, the, for, to him, my, etc.) are often appended to the content words that immediately follow or precede them. This means the number of possible “words” is much larger than in English, resulting in a large percentage of new words being outside any particular vocabulary. In addition, Arabic is spoken in a variety of dialects, which differ considerably from the formal Modern Standard Arabic that is typically used in news reports that are read by a news anchor. Thus, some percentage of the speech from field reports or informal discussions contain a certain amount of this dialect variation. Mandarin has different issues. First, unlike English and Arabic, Mandarin uses tone (different pitch and accent patterns) to distinguish different words. This means we must include features that relate to these tones. In addition, unlike Arabic which concatenates some morphemes, Chinese is normally written with no spaces between the characters at all. Thus, we don’t know, for sure, where one word ends and another begins. 3.5 and 3.6 deal with specific solutions that have been developed for dealing with each of these issues in Arabic and Mandarin. Also Chinese has many different dialects that are even more different than the Arabic dialects, the GALE Program has only used the Mandarin dialect, so the only variations that we see are due to regional accents, partially related to the native dialect of a speaker, or simply where they are from.

Finally, since the goal is ultimately to translate the speech, we must consider the effect of speech recognition systems on the translation process. For example, we must acknowledge that there will always be a significant number of speech recognition errors that the translation will have to deal with. In addition, unlike text, which usually has

relatively clear sentence breaks and punctuation, which provide additional information, speech has no such information. So a system must either make do without this information or try to recreate it. 3.7 deals with methods that have been used to optimize a translation system for use together with speech recognition systems.

Thus, the chapter is divided into seven Chapters:

1. Introduction
2. Front end feature computation
3. Modeling techniques for speech recognition
4. Language modeling for speech recognition
5. Language-specific techniques for Mandarin
6. Language-specific techniques for Arabic
7. Integration of speech recognition and translation

Each of these chapters has a separate introduction that briefly summarizes the individual sections in the section. The casual reader may find that some of the sections within a section use somewhat similar techniques, and thus, their descriptions often discuss some of the same issues. However, this apparent duplication is not accidental. It results from the fact that, as a research community, we learn from each other. When one research site develops and reports a technique that appears to work, a second site will often try to replicate that technique, first to validate the method, and then to try to find further improvements. While some might believe that this duplication of effort and similarity of approaches is wasteful, it actually is the mechanism that results in rapid progress over the years as each of us stands on the shoulders of our colleagues in order to reach higher.

3.2 Front End Features

3.2.1. Introduction

Author: Lidia Mangu

Most state-of-the-art speech recognition systems (STT) use a cepstral frontend. One of the recent trends is a multi-stream approach, in which different types of features are extracted from the speech signal and combined with the classic Mel Frequency Cepstral Coefficients (MFCC) or PLP features either at the feature-level, model level or on the output of multiple systems. Multi-layer perceptrons (MLPs) provide an effective mechanism for the transformation of alternate feature streams into a form that is both phonetically discriminative and amenable for HMM-based speech recognition. Since MLP features cover a wide temporal content, they could capture different speech properties than the cepstral features, therefore, an important question is how to best incorporate both in an STT system.

In this section, we will present the basics of the MLP-based techniques, various features of MLP, the most effective way of combining information coming from different

representation of the speech spectral dynamics and combination strategies for speech recognition systems using different acoustic frontends.

Specifically, in addition to a stream based on classic PLP features, we will show the use of streams incorporating the temporal trajectory of logarithmic spectral energy in each critical band over a long (500ms) window, and describe different combination schemes for the two streams. We will also show results incorporating a new stream based on Gabor filtering of the input spectrum.

Next, we will investigate three types of MLP features derived using a 4-layer bottleneck MLP architecture, which differ in their input speech representation. Two types of raw features are used: 9 frames of PLP based features, and time-warped linear predictive TRAP (wLP-TRAP), which are processed by the MLP. The third MLP feature set is obtained by combining the two sets of raw features at the input to the MLP. It is shown that without adaptation the MLP features perform better than the PLP features, and after adaptation they have comparable performance. Different means of combining the MLP and cepstral features are explored. The authors found that the best recipe is to combine features based on different time spans (e.g., PLP and wLP-TRAP) as they provide high complementarity, and feature concatenation is the most efficient combination method.

Combination strategies for speech recognition systems using different acoustic front ends, including the Gammatone feature, neural network based toneme posteriors and a tonal feature will be described. We will show that the best result is achieved by stacking multiple system cross-adaptation and minimum time frame error rate on the different base systems.

Finally, we will investigate the most effective ways of combining information coming from different representation of the speech spectral dynamics with a study of two types of combinations of multiple streams: one which involves training of multiple MLPs in either a parallel or sequential fashion, and another one based on frame-level combination. The study shows that combination of classifiers trained on different ranges of auditory frequencies is more effective if performed in parallel fashion, while hierarchical techniques are more effective for combining information in different ranges of modulation frequencies. For the frame-level combination, using Dempster-Shafer combination rule that closely relates to observation on how humans recognize speech in noisy conditions is proposed. The new method is shown to be effective when there is not enough in-domain training data.

3.2.2. MLP-Based Feature Extraction for Speech Transcription

Authors: Nelson Morgan, Arlo Faria, Suman Ravuri and Sherry Zhao

3.2.2.1 Introduction

Tandem acoustic feature extraction (Hermansky *et al.* 2000b) was introduced to leverage the discriminative power of a multi-layer perceptron (MLP) classifier for automatic speech transcription, producing a feature representation based on local estimates of phone posterior probabilities. Such MLP-derived features have been used for large vocabulary ASR (Morgan 2005; Zhu 2005; Zheng 2007) complementing other discriminative

methods such as Minimum Phoneme Error (MPE) parameter estimation (Povey 2002) and feature MPE (fMPE) feature transforms (Povey 2005). In our work for DARPA, first in the EARS program and then in GALE, we developed a multi-stream feature extraction module making use of the Tandem principle. In addition to a stream based on the classic Perceptual Linear Prediction (PLP) features, we also made use of streams incorporating the temporal trajectory of logarithmic spectral energy in each critical band over a long (500 ms) window. The two streams, both of which used MLPs for discriminant transformation, were combined using one of several possible weighting schemes (briefly described in Section 3.2.2.3). The combined values, which could be viewed as estimates of the posterior probabilities for phonetic classes, were then processed by a logarithm and by Principal Component Analysis (PCA) (the latter for orthogonalization and dimensionality reduction). These features were then appended to a vector of MFCCs and derivatives (that itself had been processed by Heteroscedastic Linear Discriminant Analysis (HLDA)), and the final vector was used for training and recognition of the Gaussian-mixture-based HMMs.

The above paragraph is a very brief description of techniques that have been developed over the last 20 years, starting with the interpretation of MLP outputs as posterior probabilities in work done with H. Bourlard in the late 1980's (Morgan 1995), the development of Tandem and Temporal Pattern approaches (due largely to H. Hermansky and colleagues in the late 1990's, and this decade). In two successive DARPA programs, we have done the engineering required to apply these approaches successfully to large vocabulary tasks, first for conversational telephone speech (in EARS) and more recently to Mandarin broadcast news and conversation (in GALE).

3.2.2.2 HATs and TRAPS

As noted above, a key insight from Hermansky (Hermansky and Shamma 1998) was that it should be useful to incorporate features extracted from much longer time ranges than the typical 25 ms analysis window (e.g., 500 to 1000 ms), and in particular, based on temporal properties within narrow bands (e.g., one to three critical bands). We also found that such features nicely complemented features extracted from shorter ranges (e.g., 100 ms or less). Various forms of these narrowband temporal trajectories were explored by Hermansky and his colleagues, and were called TempoRAI Patterns, or TRAPs (Hermansky and Shamma 1998). In most cases, these features were based on the energy trajectories within one or three critical bands. We found that it was useful to train critical band MLPs for phonetic classification using a single hidden layer, but then to only incorporate the learned input to hidden transformation for each band, and then to train a large MLP for phonetic classification, taking as input the hidden layer outputs for all of the critical bands. We named this TRAPs variant "Hidden Activation TRAPs", or "HATs". In later work, we found that similar results could be achieved with a single large network that was trained in a single process by having two hidden layers and constraining the connections at from the input to the first hidden layer to be isolated to local frequency regions. The latter approach has significant practical advantages, as the HATs as originally formulated require the storage and retrieval of large intermediate files. HATs

were used in most of our evaluations, but we expect to use the single “tonotopic” network for future ones.

3.2.2.3 Combining Probabilistic Features

One of the key advantages of using MLP-based feature streams is the property that properly trained MLPs can output estimates of posterior probabilities (the probability of each class given the network's input). This then suggests a number of ways for combining stream outputs. In one approach, independent streams are combined by a product rule. In another approach, streams are combined additively with gating weights. A particular method for computing these weights makes use of stream entropy estimates. Interpreting MLP outputs as a discrete posterior probability distribution, entropy is computed for each stream, and is used as an inverse confidence for the stream. The inverse entropy has therefore been used both at IDIAP and ICSI for stream combination (Morgan *et al.* 2005), as well as, a related technique derived from Dempster-Shafer theory (see Section 3.2.5) for an explanation of the latter technique). Our experience has been that both methods work reasonably well. However, when the evaluation data is reasonably well represented by the training data, another combination method is simply to train yet another network to make optimal use of both streams. Still other combination methods (for instance, training a network to generate linear stream weights) are an active area of research in our group.

While it is intellectually pleasing to generate probabilities as features, there are a number of pragmatic steps that we have sometimes found to be useful in practice. For instance, for our large vocabulary evaluations we have found it necessary to generate a log or log-like function of the probabilities, and to orthogonalize the output and reduce its dimensionality (typically with PCA). Furthermore, we have sometimes found that it is preferable to use the input to a final softmax function in the MLPs rather than the output for feature generation. This corresponds to a log probability plus a frame-dependent scale constant. This constant (which comes from the magnitude of the input to all of the output nonlinearities in a stream MLP) may contain information about the strength of each stream.

3.2.2.4 “Idealized” MLP-based Features

We have run an experiment in which the MLP was simulated to be at its optimum, providing essentially perfect classification of phonetic speech units. The outputs of the simulated classifier were replaced with forward-backward probabilities by aligning HMM models composed from reference word sequences. We then used the simulated classifier exclusively during training and applied the normal MLP on test data. Surprisingly, this mismatch did not deteriorate performance, but instead provided considerable improvement over the standard Tandem procedure. Inspired by this result, we developed a simple technique for preparing *corrected* Tandem features based on linear output activations of an MLP, and demonstrated significant improvement on a Mandarin broadcast news ASR task.

Our experiments are based on the Mandarin broadcast news ASR system described by Lei (2006), simpler than our state-of-the art implementation (Hwang *et al.* 2007)

developed as a multi-site collaboration for the DARPA GALE project. SRI's DECIPHER recognizer was configured for word-based modeling, although all ASR results are reported as character error rates (CER). The training set (Mandarin Hub4) comprised 30 hours of television shows, carefully transcribed including speaker labels. Test data are from the DARPA EARS RT-04 evaluation (eval04) and the DARPA GALE 2006 evaluation (eval06). Automatically-segmented utterances were clustered and assigned pseudo-speaker labels. Standard acoustic features were based on MFCC, warped with vocal tract length normalization and mean-and-variance normalization applied on a per-speaker basis. Since Mandarin is a tonal language, it was useful to additionally include a smoothed log-pitch estimate (Lei 2006). Adding two temporal derivatives resulted in a 42-dimensional acoustic feature vector, which we will simply reference as "MFCC". Within-word triphone HMM models were based on a 72-phone inventory comprising consonants and tonal vowels. Parameters were shared across 2000 states clustered with a phonetic decision tree, and observation distributions were modeled by a diagonal covariance GMM with 32 mixture components. Viterbi realignment of the training data was used for maximum likelihood parameter estimation. Recognition networks were compiled from trigram language models trained on over one billion words, with a 60K lexicon (Hwang *et al.* 2007). Two decoding passes were separated by 3-class Maximum Likelihood Linear Regression (MLLR) speaker adaptation, all operating in under 5 times real time.

Feature	Train	Test	Eval04	Eval06
MFCC	-	-	19.2	30.6
Tandem	MLP	MLP	15.5	24.2
Tandem	Corrected	MLP	15.1	23.9

Table 3.1: Comparison of standard and corrected Tandem features derived from an MLP's linear output (pre-softmax MLP) activations. A somewhat simplified DECIPHER system with 30 hours of training were used for these results.

We used a simple technique for correcting the linear activation outputs of an MLP. Using the Viterbi aligned reference labels for the training data, we determined for each frame whether the MLP's classification was correct. If the MLP's maximal output correctly related to the aligned phone, we left all the outputs unmodified for that frame. If the MLP's classification was incorrect, we changed the value at the output unit that should have had the maximal activation; we increased it to have the same value as the maximal activation over the other output units. This correction was a relatively minimal modification to the MLP outputs: it was applied only to frames that were incorrectly classified – about 20% of the training set – and affected just one of the MLP output units. Table 3.1 shows the experimental results using corrected Tandem features for acoustic model training. For both the eval04 and eval06 test sets, the corrected training features provided modest improvement over the features from unmodified MLP outputs. Over the two sets, the statistical significance of the systems' difference was verified by a two-tailed MAPSSWE test (Gillick 1989) $p = 0.015$ (179 vs. 135 unique errors).

The system was significantly improved by applying a small correction to the training features. The correction procedure is very simple to implement and relies only on having

Viterbi-aligned reference labels for the training data; this information is often readily available as it is typically used to prepare the one-hot encoded targets for MLP training. Some further details are given by Faria (2008).

3.2.2.5 Adaptation Methods

MLPs from one task or genre can often be adapted for another. Analogous to adaptation of probability estimators like mixture Gaussians, this can particularly be advantageous if one has an MLP trained on a large amount of data that needs to be applied to a new task or genre with less available training data. For instance, in the past we have adapted MLPs trained on thousands of hours of conversational telephone speech to an unrelated conversational meeting task (Janin 2006). More recently we have begun experimentation with genre-specific adaptation of MLP weights to reduce errors for conversational broadcast Mandarin, which occur much more frequently than is the case for Mandarin broadcast news. Show-based adaptation has been unsuccessful, but we are now investigating adaptation with a finer granularity.

3.2.2.6 Ensemble feature selection

In all of our multi-stream MLP-based approaches, we assess the value of adding a stream by including the entire stream. We do wonder, however, if the composition of each stream should change when the features in it are added to the overall pool of features used. To put this another way, would there be any advantage to put all of the features from the multiple streams into a pool and to look for the best elements in each stream? This question is sometimes referred to in the machine learning community as ensemble feature selection. While we have no answer yet for the specific tests conducted in GALE evaluations, one of the students in our group recently completed a PhD dissertation (Gelbart 2008) on his topic using a small vocabulary task (recognizing Numbers 95 utterances), both with and without significant amounts of additive noises of different types. In nearly all experiments, the rearranged multi-streams (obtained by a hill climbing procedure) performed significantly better than the original streams (which for this case were MFCC, PLP, and the Modulation Spectrogram features developed in the '90's by Kingsbury (1998)). This suggests that there may be some further optimization that may be obtained by rearranging the new features in the multi-stream combinations that we come up with for tasks like those of interest to GALE.

3.2.2.7 Parallelization to Enable More Experiments

Multi-layer perceptrons provide a useful tool for transforming new types of features into a form that is both discriminant and is applicable for use as observation variables to be modeled by Gaussian-mixture-based acoustic models. Their biggest disadvantage is the fairly long training times required. We have found that we can use sampling techniques to reduce training time significantly with only small increases in error rate, for instance by training with a reduced amount of the data with equally distributed phone priors. However, for evaluation systems, we still typically train with all of the available data, which requires a significant time to train the MLPs for each change in the feature

extraction. In earlier times our group developed our own hardware to help with this problem. In recent years, though, commercial products have become powerful enough to permit the training of very large MLPs in a reasonable amount of time. However, what is “reasonable time” depends on your goal. For evaluation subsystems, current training times of a few days to two weeks are acceptable. But, if the goal is to vary the feature methodology and test it on a day-to-day basis for research purposes, these training delays are obviously too long.

Because of the recent leveling off of clock rates in CPUs, the industry is now moving towards highly parallelized computation. One example of this is the GPU approach, with a number of products available for use as an attached processor. There can be significant difficulties with the programming of these devices, but once programmed they can provide enormous speedups for various parts of the recognition system, but particularly for the MLPs. Other products are likely to follow that might be simpler to program. In the meanwhile, we are working to parallelize our basic MLP algorithms. For instance on the Nvidia GeForce GTX280 we now are showing a significant speedup on an MLP training algorithm that was reprogrammed from our legacy code for better parallelization. This work is being assisted by our collaboration with the Berkeley Parallel Lab (Asanovic 2008), an industry-funded effort to crack the problem of writing software for new parallel (“manycore”) architectures.

3.2.2.8 Conclusions

Over the years, we have found discriminatively trained features, using MLPs, to be an effective augmentation of the more standard observation vectors used for automatic speech recognition. They have now become a standard component in the systems developed for programs like GALE. However, while progress has been made, a number of issues remain open, such as: optimal combination of MLP outputs; training of HMMs using blends between actual and idealized MLP outputs; adapting MLPs for differing speech genres (e.g., read news vs. conversation); incorporating newer features that are more biologically inspired; rearranging features in streams for optimal effect; and radically speeding up training and test cycles using newer hardware/software architectures so that many more experiments can be run. Finally, although this point is not the focus of the current section, we feel that incorporating performance measures from later stages in the overall task (e.g., machine translation errors) should ultimately be used for design choices in the speech recognizer, which will require a tighter coupling than is currently feasible.

Acknowledgment: Although many colleagues at ICSI, SRI, and IDIAP have helped us in this work over the years, we want to particularly acknowledge the assistance of Andreas Stolcke, without whom we never would have been able to run our large vocabulary ASR experiments; and Hynek Hermansky, who was a prime contributor to many of the basic ideas that we took advantage of.

3.2.3. Combining MLP and PLP Features for Speech Transcription

Authors: Petr Fousek, Lori Lamel and Jean-Luc Gauvain

3.2.3.1 Introduction

One of the recent trends in speech recognition is using discriminative techniques with large corpora for more accurate acoustic modeling. Maximum likelihood training of Gaussian mixture HMMs is often replaced by Maximum Mutual Information (MMI), Minimum Classification Error (MCE), or MPE criteria and features are being modified by discriminatively trained transforms such as feature-level MPE (Povey 2005). There has been growing interest in incorporating some discriminative estimation in the feature extraction by using discriminative classifiers such as MLP. Since MLP features cover a wide temporal context, they can potentially capture different speech properties than are captured by the widely used cepstral features. In addition, MLPs can be trained to deliver estimates of class posteriors which can be directly used as emission probabilities in Hidden Markov Models. Over the years, ICSI, SRI, UW and other groups have developed mature techniques for extracting probabilistic MLP features such as TRAPs, and have substantial experience incorporating these MLP features in speech-to-text systems (Zhu 2005, Stolcke 2006). While probabilistic features have never been shown to consistently outperform cepstral features in LVCSR, they have been shown to improve performance when used in conjunction with them. The MLP and PLP features being complementary, an important consideration is determining the best manner to incorporate both in an STT system. Different means of multi-stream combination have been successfully used for this purpose, four of which were studied in Fousek (2008a).

3.2.3.2 Task and System Overview

The speech recognizer was derived from the LIMSI Arabic STT component system used in the Agile participations in the Gale evaluations. The transcription system has two main parts, an audio partitioner and a word recognizer (Gauvain 2002). The recognizer makes use of continuous density HMMs for acoustic modeling and n-gram statistics for language modeling. Each context-dependent phone model is a tied-state left-to-right CD-HMM with Gaussian mixture observation densities. Word recognition is performed in one or more passes, where each decoding pass generates a word lattice with cross-word acoustic models, followed by consensus decoding with 4-gram language model (LM) and pronunciation probabilities (Gauvain 2002; Lamel 2007). Unsupervised acoustic model adaptation is performed for each segment cluster using the Constrained MLLR (CMLLR) and MLLR (Leggetter 1995) techniques between decoding passes.

The manually transcribed Arabic broadcast news (BN) and broadcast conversation (BC) data distributed by the Linguistic Data Consortium were used to train the acoustic models. There are over 1380 hours of raw data, with roughly 730 hours BN and 550 hours of BC. After removing non-speech portions (music, publicity) and portions that fail forced alignment, about 1250 hours of data remain for HMM training. This is referred to as the 1200 hour training set, used to train the baseline gender-independent

acoustic models, covering 44k word position-dependent contexts with 11.5k tied states (32 Gaussians/state) (Fousek 2008). Some results are reported for a development system, using 300 hours of data for acoustic model training.

Various language models were trained on corpora comprised of 11M words of audio transcriptions and 1B words of texts from a wide variety of sources. The language models result from the interpolation of models trained on subsets of the available data, with the interpolation weights optimized on the combined GALE development data from 2006 and 2007. The coefficients associated with the audio transcriptions are assigned almost half the LM weight, even though these texts represent only about 1% of the available data. Language models were estimated on the normalized texts and morphologically decomposed texts (Lamel 2008). For multi-pass decoding, lattices are rescored by a neural network LM (Schwenk 2007) interpolated with a 4-gram back off LM. Results are reported for several GALE data sets, where each set contains about three hours of broadcast data.

3.2.3.3 Training MLP Features

The neural network feature extraction has two steps. The first step is *raw feature extraction* which constitutes the input to the MLP. Typically this vector covers a wide temporal context (100-500ms) and, therefore, is highly dimensional. Second, the raw features are processed by the MLP followed by a PCA transform to yield the HMM features.

Raw Features

Two sets of raw features are used which cover different temporal contexts: 9 frames of PLP (9xPLP) and *time-warped linear predictive* TRAP (wLP-TRAP) (Fousek 2007). The 9x PLP set is based on the PLP features from the baseline system, which are mean and variance normalized per speaker. To the best of our knowledge, this was the first time that the wLP-TRAP features were incorporated in a state-of-the-art system. Since the MLP topology, as well as the speech representations at the MLP input used here differ from the better known ones, it is of interest to explore the properties of these features and suitable ways of combining them with cepstral ones.

ID	Raw features (#)	HMM features (#)
PLP	PLP(13)	PLP+ Δ + Δ^2 (39)
MLP _{9xPLP}	9x(PLP+ Δ + Δ^2)(351)	MLP(39)
MLP _{wLP}	wLP-TRAP(475)	MLP(39)
MLP _{COMB}	9x(PLP+ Δ + Δ^2)+ wLP-TRAP(826)	MLP(39)

Table 3.2: Naming conventions for MLP features and how the raw input features relate to the features for HMM

The raw features are formed by 9 neighboring frames of PLP (12 coefficients plus energy, with derivatives Δ and Δ^2), centered at the current frame. The feature vector has $9 \times 39 = 351$ values and covers a 150ms window. The wLP-TRAP raw features are obtained by warping the temporal axis in the LP-TRAP feature calculation. Linear prediction is used to model the Hilbert envelopes of 500~ms long energy trajectories in

auditory-like frequency sub-bands (Athineos 2004). 25 LPC coefficients in 19 frequency bands form the raw features, yielding $19 \times 25 = 475$ values which cover a 500~ms window. wLP-TRAPs use a different bank of filters than are used for the PLP and they do not apply a short-term FFT, so they have a potential of producing more complementary features to PLP than 9-PLP or TRAPs, which should be an advantage for feature combination. The adopted naming conventions for the raw features and the HMM features derived from them are given in Table 3.2. The last feature set is obtained by combining the 9xPLP and the wLP-TRAP features at the input to the MLP and produces a 39-parameter feature vector.

MLP Architecture

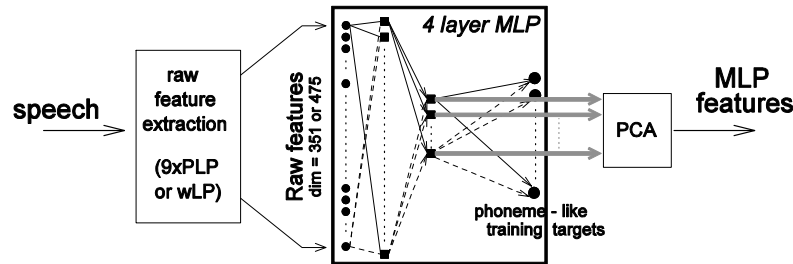


Figure 3.1: MLP feature computation with a four layer bottleneck network, using phoneme-like training targets.

The MLP architecture is based on a four layer bottleneck network with an input layer, two hidden layers and an output layer, as shown in Figure 3.1.

The input layer distributes the raw features in the second layer, which is large in order to provide the necessary modeling power. The third layer is small, its size is equal to the required number of features, which in this work was fixed to 39 for easy comparison with PLP features.¹ The output layer computes the estimates of the target class posteriors. The classes are context independent phone states obtained from a HMM automatic alignment which were shown to outperform phone targets (Grézl 2008). There are 69 three state phones and 3 units with all states merged (silence, filler, breath) resulting in 210 target classes. The outputs of the small hidden layer neurons (prior to a sigmoid function) are decorrelated by a PCA transform and used as final features. Note that this MLP architecture allows the feature vector size to be arbitrarily chosen, independently of the number of MLP targets.

MLP Training Data

The MLP features were trained on the 1200 hour train set. Since the MLP features make use of a temporal context up to 500ms, the frames from first and last 250ms of each segment are not used for training (except for providing a context), thus the data available to train the MLP is 1168 hours. It is known that more data and/or more parameters in the MLP help, but at certain point the gain is not worth the effort. Table 3.3 gives the word error rate as a function of the amount of MLP training data for a MLP with a fixed

¹ The optimal number of features was explored by Grézl (2008).

number (1.4 million) of parameters with 9xPLP raw features. The MLP performance is seen to improve with the additional data, and no saturation is observed.

MLP train set	17 hrs	63 hrs	300 hrs	1200 hrs	PLP baseline
bnat06 WER %	24.7	24.2	23.4	22.2	25.1

Table 3.3: Word error rates on the bnat06 data set as a function of the amount of data use to train the MLP_{9xPLP}. All HMMs trained on 300 hours of speech. Single decoding pass with 200k gram LM, no adaptation, no MLLT,² no MMIE.³

Training Process

Training a MLP on over thousand hours of speech required two modifications to the training process performed by QuickNet software.⁴ First, the storage space requirements of the raw features were reduced by almost a factor of four by using linear quantization of 32 bit float values to 8 bits, with no impact on performance. Once the MLP is trained, output features are created using non-compressed raw features. Second, to reduce the computation time of MLP training, a simplified training scheme was adopted after (Zhu 2005). Instead of iterating on all training data about 10 times through the MLP as determined by cross-validation performance, a fixed number of 6 epochs with fixed learning rates is used with subsets of the data. The data are randomized and split in three non-overlapping subsets of 13%, 26%, and 52% of the frames. First, three epochs are trained on 13% of the data, then two subsequent epochs use 26% of the data, the last epoch uses 52% of the data, with the remaining data used to monitor the performance. This reduced the training time by a factor of 5.4 with a negligible impact on performance (in fact, tests with the 300 hour set even improved from 24.4% to 24.2% WER on the bnat06 data for the 9xPLP raw features, with unadapted models). All these modifications reduced the training time to about one week on the 1200 hour train set using one four-threaded computer, and the wLP-TRAP) raw training features occupy 200GB of disk space.

MLP Size

To get the most benefit from the larger amount of training data may require using a more complex model. An experiment was carried out by enlarging the first hidden layer in the MLP in order to raise the number of free parameters. The results show that for the small 63 hour train set, the larger MLP degraded performance, while for the full 1168 hours it brought a 1.6% relative improvement. However, such a gain was not judged to be worth the computation cost (almost four times longer to train the MLP), so further experiments used the smaller MLP with 1.4 million parameters.

² Maximum Likelihood Linear Transform

³ Maximum Mutual Information Estimation

⁴ <http://www.icsi.berkeley.edu/Speech/qn.html>

3.2.3.4 Using MLP Features

This section presents contrastive results starting with the baseline system, and going to more complex models and decoding strategies.

Experiments with a Small System

A first series of experiments were carried out to compare the four basic features from Table 3.2 without and with unsupervised acoustic model adaptation. The 1-pass results use a 4-gram LM, no adaptation, no MLLT, no MMIE. The MLPs were all trained on 63 hours of data, and all HMMs were trained on 300 hours of speech. The baseline performance of the standard PLP features without adaptation is 25.1% and with adaptation is 22.5%. Without adaptation, the $MLP_{9 \times PLP}$ features are seen to perform a little better (about 4% relative) than PLP, but with adaptation both $MLP_{9 \times PLP}$ and MLP_{wLP} are slightly worse than PLP. This leads to the conclusion that MLLR adaptation is less effective for MLP features than for PLP features.

Different means of fusing the information coming from the cepstral and the MLP features were investigated. The MLP_{comb} results are for combination at the input to the MLP. These models give the best results with 39 parameters. A simple approach is to concatenate together the features at the input to the HMM system (this doubles the size of the feature vector, $2 \times 39 = 78$ features) and to train an acoustic model. Three possible pairwise feature concatenations were evaluated. These concatenated features all substantially outperform the PLP baseline, by up to 9% relative, showing that feature concatenation is a very effective approach. Given the significantly better performance of the $PLP+MLP_{wLP}$ features over the $PLP+MLP_{9 \times PLP}$ and $MLP_{9 \times PLP}+MLP_{wLP}$ features, the three-way concatenation was not tested as it was judged to be not worth the increased computational complexity needed to deal with the resulting feature vector size 3×39 .

Two other more computationally expensive approaches were studied, cross model adaptation and ROVER (Fiscus 1997). Adapting the PLP models with the hypotheses of the MLP_{wLP} system, or the reverse adaptation using the hypotheses of the PLP system for MLP_{wLP} produces a 3.6% and 3.1% absolute improvement to the PLP only system (25.1% WER). Such cross adaptation using concatenated features such as $MLP_{9 \times PLP} + MLP_{wLP}$ produces 4.4% and 3.9% absolute improvement. Larger gains are obtained when the PLP models are used in the second pass, supporting the earlier observation that MLLR adaptation is more effective for PLP features than for MLP features. This may be because the MLP already removes the variability due to the speaker or because other, perhaps non-linear, transformations are needed to adapt MLP features.

	Combined systems	$PLP \otimes MLP_{9 \times plp} \otimes MLP_{wLP}$	$PLP \otimes MLP_{wLP}$	$(PLP+MLP_{9 \times plp}) \otimes (PLP+MLP_{wLP})$	$PLP \otimes (PLP+MLP_{wLP}) \otimes (MLP_{9 \times plp}+MLP_{wLP})$
Wer (%)	1-pass	22.3	23.3	21.2	21.0
	2-pass	20.6	21.0	19.9	19.7

Table 3.4: Comparing ROVER for combining multiple systems on bna06 data (PLP only produces 25.1% WER).

The WERs resulting from ROVER combination is shown in Table 3.4. As is shown in the table, ROVER combination of the three basic features performed better than the best pair-wise cross-adaptation amongst them however, neither combination outperformed the simple feature concatenation of PLP+ MLP_{wLP} with a WER of 20.4. ROVER also helps when applied jointly with other combination methods (see the last two columns in Table 3.4), beating the baseline PLP system by up to 5.4% absolute (12% relative). This best result however requires 6 decoding passes.

It is interesting to observe that the PLP features are generally best combined with MLP_{wLP} , even though the $MLP_{9 \times PLP}$ gives better score than MLP_{wLP} . This may be due on one side to the fact that the $MLP_{9 \times PLP}$ features are derived from the PLP, and on the other side that there is a larger difference in time spans between the standard PLP and the wLP-TRAP features.

Amount of MLP Training Data

We can compare performances of the MLP features when used stand-alone and when concatenated with PLP features at the input to the HMM system as a function of the amount of data used to train the MLP. Note that the concatenated vector has 78 features, whereas the stand-alone vector has 39 features. The HMMs were all trained on the 300 hour data set. For all feature sets there is a significant WER reduction when the MLP training data is increased from 63 to 300 hours. The results with the two types of MLP features stand-alone are comparable when 300 hours are used to train the MLPs. HMMs trained with both MLP features outperform the PLP baseline (25.1%). Concatenating the PLP features with the MLP ones gives the best performance, however, the improvement from training the MLP on more data is less than for the systems using only MLP features. The best results are obtained with the HMM trained on the PLP+ MLP_{wLP} features (21.7% for 63 hours and 21.3% for 300 hours).

A system with improved 290k 4-gram LM, improved pronunciation modeling and gender-dependent models was tested. The PLP-based system also has MLLT. Training the both the HMM and the MLP aspects of the system on 1200 hours of speech has reduced the WER further to 19.2% for the PLP+ MLP_{wLP} features.

Using standard techniques of Speaker Adaptive Training (SAT), CMLLR and MLLR adaptation shows that without adaptation the MLP_{wLP} and concatenated PLP+ MLP_{wLP} features clearly outperform the PLP ones. However, with adaptation, only the concatenated features perform significantly better than the PLP, bringing the WER to 17.8%.

The last set of experimental results were produced with a more complete system, including gender-dependent SAT, MMIE acoustic models with word duration models and MLLT for PLP, trained on 1200 hours of manually transcribed data. It uses a multiple pass decoding with CMLLR and MLLR adaptation, a word- or morph-based 290k 4-gram neural network (NN) language model, and improved pronunciation models. NN LM results from the interpolation of a connectionist language model with a standard 4-gram back off LM.

Acoust. Models	PLP	MLP _{wLP}	PLP+ MLP _{wLP}	PLP⊗(PLP+MLP _{wLP})	PLP	PLP+ MLP _{wLP}	4-way ROVER Combination
Lang.	word	word	word	word	morph	morph	both
Models							
bnat06	16.7	16.8	15.4	15.0	16.7	15.7	14.5
bcat06	22.8	22.7	21.1	20.7	23.2	21.9	20.2
eval06	13.5	14.3	13.0	12.4	13.8	12.9	11.9

Table 3.5: WER on the broadcast new (BN) and broadcast conversations (BC) data sets from '06 and the evaluation set from 2007. The acoustic models are gender-dependent SA, MMI trained PLP and MLP models (also with MLLT for PLP) trained on 1200h of manually transcribed data, with word duration models. Multiple pass decoding with CMLLR and MLLR adaptation, a 290k 4-gram NN LM, and improved pronunciation models. Also show are results for 2-way and 4-way ROVER combinations.

Table 3.5 gives the word error rates for three acoustic models (PLP, MLP_{wLP} and PLP+ MLP_{wLP}) for the BN and BC '06 and evaluation '07 data sets. It can be seen that the PLP and MLP_{wLP} based systems give comparable results. Based on the combination experiments reported by Fousek (2007), we selected a 2-way ROVER combining the PLP and the PLP+ MLP_{wLP} based systems, which gives a gain of almost 0.5%. The results with the morphologically decomposed LM (Lamel 2008) are seen to be comparable to those with the word-based LM. A 4-way combination gives an additional 0.4% gain over the 2-way ROVER.

The performance of the PLP system has been improved from the baseline of 25.1% to 16.7% on the bnat06 data set (a relative WER reduction of 33%). The combined PLP+MLP_{wLP} based system is seen to obtain a lower WER for all test sets, with an average gain of 1.2% absolute. ROVER combination of the PLP and PLP+MLP_{wLP} based systems gives a gain of over 4% absolute, even though the PLP features are in there twice.

3.2.3.5 Summary

This section has explored incorporating novel MLP features, derived using the bottleneck MLP architecture, in a state-of-the-art Arabic broadcast data transcription system. In particular the influence on performance of the amount of data used to train the MLP, the number of free parameters in the MLP, and the amount of data used for HMM training was assessed. Different means of combining MLP and cepstral features were also explored. Experiments were carried out on the GALE Arabic broadcast task using multiple data sets. When used without adaption, the MLP features have better performance than standard PLP features. However, once SAT training and CMLLR/MLLR adaptation are used, both feature types have comparable performance. Feature concatenation appears to be the most efficient combination method, providing the best gain at the lowest decoding cost. It seems best to combine features based on different time spans as they provide high complementarity. Since the PLP based system improves more than the MLP based with unsupervised adaptation, an additional gain is obtained by combining a PLP based system with one based on the concatenated features and with ROVER combination using different language models. It also seems that gains from MMI model training are additive to the gain coming from discriminative MLP features.

Recently, speech recognizers have been trained using the combined PLP+wLP-TRAP features for broadcast data transcription in Dutch, Flemish, French and Mandarin, and have observed comparable system behaviors and performance gains. An attempt was made to use an MLP developed for one language to produce features for another, but this

was not successful, suggesting that the MLP features are capturing language-specific information. These features were also found to improve performance for the transcription of conversational telephone speech in Dutch and Flemish (Despres 2008).

One of the difficulties in carrying out such experiments with a full system is that generating the time-warped linear predictive TRAP features and training the MLP are quite computationally expensive. Some initial experiments have been carried out with other features, Multi-RASTA (MR) and TRAP-DCT (TD), that are much less costly to obtain (Hermansky and Fousek 2005). The TRAP-DCT features are obtained from a 19-band Mel scale spectrogram, using a 30 ms window and a 10 ms frame step. A discrete cosine transform (DCT) is applied to each band, resulting in 475 raw features, which are fed to a 4-layer MLP (bottleneck architecture). Using the small training, the MLP_{TD} features give a word error rate of 24.4% alone and 21.4% when combined with PLP features on the bnat06 data set, which are comparable to results for the MLP_{wLP} features. With full training on 1200 hours, both feature sets obtain comparable results (13.7% PLP+ MLP_{wLP} and 3.6% PLP+ MLP_{TD}), outperforming standard PLP features (14.5%) on the dev08 data. This result is very interesting since the raw TD features are not much more costly to compute than the PLP ones.

3.2.4. Multiple Feature Speech Recognition

Authors: Christian Plahl, Björn Hoffmeister, Ralf Schlüter, and Hermann Ney

3.2.4.1 Introduction

Recently, new acoustic features, such as neural network based phoneme posterior probabilities, and features based on gammatone filterbanks, have been developed for automatic speech recognition (ASR).

While we continue to investigate new features, the best way to combine different feature streams remains an open question. In the last years several approaches for combining acoustic features have been proposed. Schlüter (2006) does combination done explicitly on the feature level and Zolnay (2005) does it in an acoustic rescoring framework. Even though both approaches achieve reasonable improvements, system combination seems to be superior (Zolnay 2006). System combination can be performed on different levels. Implemented in the model adaptation step of the system it is referred to as *cross adaptation* and proved to give considerable improvements, (Guiliani 2006). Alternatively, lattice or n-best list based system combination is applied to the final output of the individual systems (Evermann 2000).

We show results on each of the three levels and how the different levels will be influenced by each other when the levels are stacked. In addition, we introduce vocal tract length normalization (VTLN) for Gammatone (GT) features. Gammatone features have been shown to be competitive and complementary to standard features used in large vocabulary speech recognition (LVCSR) (Schlüter 2007).

3.2.4.2 Acoustic Modeling

As described by Hoffmeister (2007) and Plahl (2008), we investigate three systems for a Mandarin speech recognition task differing only in the acoustic frontend. The acoustic training of these systems is performed independently for each system. The toneme set consists of 69 tonemes, a single garbage phone, and silence. The vocabulary size of the LM is 60K. The full 4-gram LM is used in lattice rescoring only, while a pruned version is applied in all other recognition steps. Overall, we use a 3-pass recognition procedure, where the first two passes are realized by a 4-gram Viterbi decoder, while the third pass uses lattice based LM rescoring.

3.2.4.2.1. Acoustic Features

The different systems use the output of the MFCC, PLP, and Gammatone (GT) frontend as base features. A detailed description of the GT features is given in Section 3.2.4.2.2. The base features are normalized by segment-wise mean and variance normalization and fed into a sliding window followed by a linear discriminant analysis (LDA).

In addition, for each system we produce a tonal feature as described by Lei (2006). Tonal information is crucial for Mandarin ASR systems, because tonal patterns play an important role in distinguishing tonemes and words in the Mandarin language. The most common approach to integrate the tonal feature is a simply concatenation to the base features. In this section a LDA transformation followed by a dimension reduction step is used to improve this approach to feature combination.

In order to build the system used in the last GALE evaluations, we have combined these features with two kinds of neural network (NN) based phoneme posterior features, described in Section 3.2.3 of this book. Neural network posterior features provide discriminative phonetic information at frame level. Large performance improvements have been reported for systems integrating the NN features (Valente 2007; Plahl 2008). The NN features are used for the evaluation systems only.

3.2.4.2.2. Gammatone Cepstral Features

Gammatone (GT) features (Schlüter 2007) have been shown to be competitive to standard features like MFCCs or PLPs. We have tested two different GALE data sets (dev07 and eval06) and have shown that a system with unwarped GT features achieves the same character error rate (CER) as the systems using MFCCs or PLPs.

GT features are extracted by auditory filter banks realized by Gammatone filters. Originally, Gammatone filters were introduced by Aertsen (1980).

The Gammatone filter is defined in the time domain by the following impulse response:

$$h(t) = k \cdot t^{n-1} \exp(-2\pi \cdot B \cdot t) \cdot \cos(2\pi \cdot f_{ct} + \varphi) \quad (3.1)$$

Here, k defines the output gain, B defines the bandwidth, n is the order of the filter, f_c is the filter's center frequency, and φ the phase. In this work, 4th order Gammatone filters are used.

The center frequencies of the filters are distributed over the frequency range according to the Greenwood function with human parameters (Greenwood 1990)

$$\rho(x) = A \cdot (10^{ax} - k) \text{Hz}, \quad (3.2)$$

where $A=165.4$, and $a=0.88$, and $k=2.1$.

The absolute values of the 68 Gammatone filter outputs for a sampling rate of 16 kHz are temporally integrated using a 25 ms wide Hanning window and a 10 ms frame shift. The spectral integration is followed by a 9-channel window, a 4-channel shift and a 10th root compression. Finally, the 15 cepstral coefficients are decorrelated and normalized by mean and variance.

3.2.4.2.3. VTLN for Gammatone Features

Classical acoustic features like MFCCs or PLPs are normalized by vocal tract length normalization (VTLN). In order to transfer VTLN to Gammatone features, the center frequencies of the filters have to be scaled explicitly. Similar to the VTLN implementation for MFCC and PLP a piecewise-linear transformation function is used.

After scaling the frequency space using the Greenwood function, the center frequencies are equally spaced as described in the previous Section.

In order to get the warped center frequencies, the following steps have to be applied:

- transform the minimum (100 Hz) and the maximum (7500 Hz) center frequencies to obtain the minimum and maximum warped greenwood scales:

$$[x_{min} x_{max}] = \rho^{-1}([f_{min} f_{max}]), \quad (3.3)$$

where x_{min}, x_{max} are the values of the warped frequencies f_{min} and f_{max} .

- transform the equally spaced greenwood values back to the on-warped frequency axis,

$$f_{center} = w^{-1}(\rho(x)), \quad (3.4)$$

where f_{center} defines the center frequencies of the Gammatone filters in Hz, $x \in [x_{min}, x_{max}]$ and w the warping function.

Unlike VTLN for MFCCs or PLPs, VTLN for GT features is carried out in the time domain, not in the frequency domain. Due to that, VTLN for Gammatone is done by warping the center frequencies of the Gammatone filters, instead of a redistribution in the frequency domain. As previously stated, all systems perform equally, achieving a character error rate (CER) of 17.8% on dev07 and 24.9% on eval06. When VTLN is applied to the GT features, an improvement of about 0.3% absolute for dev07 and 0.2% for eval06 is achieved.

In contrast to VTLN for MFCC or PLP, we have normalized the GT features by applying a 2-pass VTLN decoding. The warping factors are estimated by a grid search in the range of 0.8-1.2.

Since every filter's center frequency can be modeled independently, we have investigated to further improve VTLN in Gammatone feature extraction.

3.2.4.2.4. Acoustic Training

The basic setup of the systems used in this section is described by Hoffmeister (2007) and (Plahl 2008). The acoustic models are 3-state left-to-right HMMs based on triphones with cross-word context.

For all systems the baseline features are concatenated with the tonal feature. In order to compensate for speaker variations, SAT/CMLLR is applied in training and recognition.

The systems trained for the GALE 2008 evaluation use the modified MPE criterion in the discriminative training step. Modified MPE is described in Section 3.3.4.2.7 of this book.

3.2.4.2.5. Experimental Setup

In order to carry out systematic experiments we use a reduced corpus of 230h of Mandarin speech data, described by Plahl (2008), to train the models. The complete corpus contains up to 1,600h of broadcast news (BN) and broadcast conversation (BC), is used to train the evaluation systems only. The two different corpora are used to run several detailed experiments and to verify the results on a bigger amount of data. The systems are based on the MFCC or PLP frontend and tonal and neural network posterior features.

The GALE 2007 development corpus (dev07) and the GALE 2006 evaluation corpus (eval06) are used for tuning and testing the systems.

3.2.4.3 Multiple feature speech recognition

3.2.4.3.1. Integration of Acoustic Features

Several approaches to integrate acoustic features are known. We show results integrating the tonal and the NN features, trained on 1600h. The simplest approach to integrate the tonal feature, as proposed by Lei (2006), concatenates the tonal feature and the first and the second derivatives to the LDA transformed base features. Here, we present results where the base feature and the tonal feature streams are concatenate first, followed by a LDA, continuing work presented by Hoffmeister (2007).

Table 3.6 shows that the integration of the tonal features by the LDA approach slightly improves the recognition results and we see that the improvements are consistently better for all systems. Only for the PLP based system the improvement does not transfer to the test set. Nevertheless, the LDA based approach can improve the recognition performance compared to the simple concatenation of the tonal features.

	combination method	CER (%)	
		dev07	eval06
MFCC + tone	Concatenation	14.6	20.7
MFCC + tone	LDA	14.3	20.3
PLP + tone	Concatenation	14.6	20.9
PLP + tone	LDA	14.2	20.9
GT + tone	Concatenation	15.2	21.8
GT + tone	LDA	15.0	21.6

Table 3.6: Combination of the tonal feature and different feature streams based on MFCC, and PLP, and GT front-ends after SAT/CMLLR and LM rescoring.

While the choice of the feature integration method is important to improve the recognition performance, integration of additional features and perfecting these features can also help to improve the system. As shown by Hoffmeister (2007), neural network posterior features improve the system performance by about 5-10% relative. Table 3.7 summarizes the results for the two different systems build up for the GALE Re-Eval 2007 and GALE Eval 2008. In contrast to all other experiments the systems are trained on 1600h of audio data.

System	dev07 CER (%)	
	MFCC	PLP
Re-Eval 2007	11.7	10.5
Eval 2008	9.7	10.0

Table 3.7: Results for the GALE Re-Eval 2007 and Eval 2008 system using MFCCs and PLPs as base features, concatenated with tonal and NN posterior features.

The neural network features used by the Eval 2008 PLP system are a slightly modified version of the features used in the previous evaluation. In contrast, the neural network features of the MFCC system have completely changed. More information about the new neural network features can be found in Section 3.2.5 of this book. Another major difference is that the Eval 2008 systems use modified MPE, described in 3.3.4.2.7, whereas the Re-Eval 2007 system uses standard MPE. Along with some lesser changes this explains the performance differences between the Re-Eval 2007 and the Eval 2008 systems.

3.2.4.3.2. Cross-Adaptation and Lattice Combination

The results of Table 3.6 of the three different systems can now further improved by applying system combination. Here, we investigate two common combination approaches: cross adaptation and minimum frame word error combination (min.fWER) on lattices. Finally, both approaches can be stacked resulting in a further improvement.

Table 3.8 summarizes the cross-adaptation, min.fWER, and stacked combination results. In cross-adaptation we adapt to the two remaining systems using the averaged adaptation statistics of the individual systems. For the multiple supervisor cross adaptation method, marked as adapt in Table 3.8, the CER of the single systems is improved by 0.4% to 0.8% absolute for dev07 and by up to 1.2% for the eval06 corpus.

System	CER (%)			
	dev07		eval06	
	none	adapt	none	Adapt t
s1=MFCC	14.6	14.2	20.7	19.9
s2=PLP	14.6	13.8	20.9	19.9
s3=GT	15.2	14.7	21.6	20.4
s1+s2	13.6	13.6	19.2	18.8
s1+s3	13.6	13.5	19.2	18.9
s2+s3	13.6	13.2	19.2	18.9
s1+s2+s3	13.2	13.0	18.9	18.7

Table 3.8: Results for min.fWER and for cross-adaptation using two supervisors. Systems are based on MFC, PLP, or GT frontends after SAT/CMLLR and LM rescoring.

The lattice-based system combination method applied (min.fWER) is described by Hoffmeister (2006). The combinations also have been performed using ROVER and confusion network combination (CNC), achieving similar or slightly worse results compared to the min.fWER method. The aim of system combination is to minimize the expected error rate given multiple systems outputs. Bayes decision rule with a Levenshtein cost function L provides the general framework. Replacing the Levenshtein cost function with a frame-wise word error cost function leads to the min.fWER method, with a resulting algorithm of low computational complexity.

There is a substantial difference between CN and min.fWER decoding. In CN decoding word boundaries are only used to align words. Once the CN is built, all time information is lost and the calculation of the word posterior probabilities depends only on the resulting word positions. Time boundaries for the output have to be produced in a post-processing step. The min.fWER decoding approach preserves the word graph structure, and thus, the output is produced with correct word boundary times.

In order to compare cross-adaptation and system combination we have combined the output of the different systems before and after adaptation, as shown in Table 3.8. Using min.fWER reduces the error rates for all references, but the gain for the adapted systems are not as high as for the unadapted systems. Combining all three systems improves the best combination result of two systems by about 0.2% absolute for both corpora.

Overall, the best result is achieved when all three systems are used for system combination, instead of two, and the different levels are stacked.

3.2.4.4 Conclusions and Future Work

In this Section, we did a comparison of several feature integration and system combination methods. While the right feature combination method, such as LDA, could improve the system performance, the choice of additional features have also an effect of the system performance. Moreover, our experiments show that adding neural network posterior features improves the systems significantly and reduce the CER by more than 10% relative.

Furthermore, we introduced vocal tract length normalisation for the Gammatone features by explicitly modifying the center frequencies of the Gammatone filters. We showed that VTLN for Gammatone improves the system performance and achieves

almost comparable results to VTLN for MFCC or PLP features. We expect further improvements with VTLN for Gammatone features by examining other warping function.

In addition, we conducted several system combination experiments. The best result is achieved by stacking multiple-system cross-adaptation and min.fWER lattice combination of the different base systems. This combined technique reduces the character error rate of the best single systems by 5% to 10% relative. Further experiments to combine all of the levels will be performed.

3.2.5. Data-Driven Extraction of Spectral-Dynamics Based Posterior

Authors: Fabio Valente and Hynek Hermansky

3.2.5.1 Introduction

Typical Automatic Speech Recognition (ASR) features are obtained through short term spectrum of 20-30 ms segments of speech signal. This representation only extracts instantaneous frequency components of the signal. Power spectrum is then integrated using a bank of filters equally spaced on an auditory scale (e.g., Bark scale), thus obtaining the *auditory spectrum*. Spectral dynamics is generally introduced using dynamic features (temporal differentials of the spectral trajectory).

There are two principles that differentiate our work from the conventional feature extraction as described above. These principles are: 1. use of relative long (of the order, or syllable, or more) segments of the signal for extraction of instantaneous features, 2. pursuing classification of speech sounds in several parallel streams, and fusing the classifier outputs to obtain the final set of features for the subsequent HMM recognition. These principles follow an earlier proposal for cross-time processing of Allen (Allen 1994) as put to practice by Hermansky and Shamma (1998).

Longer temporal intervals for obtaining the evidence about the underlying speech sounds

To our knowledge, this has been first proposed by Kozhevnikov and Chistovich (Kozhevnikov 1967). Consistently with this proposal, and arguing that it is hard to distinguish between signals with different temporal structure (such as speech and stationary noises) from short segments of the signal (Hermansky 1994) proposed filtering of temporal trajectories of speech features by filters with relatively long impulse responses. Later, to extract information about underlying speech sounds coded in coarticulation patterns, (Hermansky 1998) formed inputs to phoneme classifier from 1001 ms long segments of temporal patterns of spectral energies in critical sub-bands. Most recent evolution of this principle is MRASTA (Hermansky and Fousek 2005) that is pursued in the work described here.

Multi-stream recognition

Fletcher and his colleagues (Fletcher 1995) were to our knowledge first to provide formal experimental evidence that human perception extracts information in multiple streams, fusing the evidence from these streams to arrive to final decision about the

perceived event. Working with information streams formed by sub-dividing the speech spectrum in a number of frequency sub-bands, they also proposed a model that accounted for their experimental data in which probabilities of errors in the individual streams multiply to yield the probability of the overall error of the classification. The model has been recently further verified (Li 2009).

The model accounts for robustness of human perception where the low-error information streams dominate the outcome. The model appears not to be limited to sub-band information streams, but also to other intra and inter modal streams of human perception. It implies conditional independence of errors in the individual information streams and requires that the human information processing chain identifies correctly recognized items, thus eliminating false alarms, so that the only errors come from the lack of sufficient information for making a decision (Pavel 2009). Intuitively, this seems reasonable.

Among first engineering system aiming at emulating Fletchers observations was the multi-band recognizer (Hermansky *et al.* 1996), later extended to multi-stream with streams formed by different parts of the modulation spectrum (Tibrewala 1997; Shamma 1999). The multi-stream recognition, where a new technique is combined with the older existing one (e.g., (Fiscus 1997)) is now one of the dominant techniques for improving the recognition performance.

There are three main issues that need to be addressed to emulate the Fletchers model:

- How to form the information streams.
- How to process signals in the individual streams.
- How to fuse information from the individual streams for the final decision.

In our first attempts, we formed the streams by dividing carrier frequency of speech into several sub-bands, and processed signals in the sub-bands either by classifying them into phoneme-like classes using multi-layer perceptrons (MLPs) or by projecting the time-frequency planes in the sub-bands on a set of linear basis. Several information fusing strategies were tried (Tibrewala 1997), one of more successful being training a nonlinear MLP-based classifier to make the final decision on a training subset of the data.

In this work, we propose forming the streams by dividing the modulation spectrum of speech into two bands, and classifying data in the individual streams into phoneme-like classes by multi-layer perceptrons. In several schemes, we apply the MLP-based information fusion.

As for techniques for the information fusion, it is fair to say that as of now, it is still not entirely clear how to design an engineering system that would emulate the Fletchers data. The work on information fusion based on Dempster-Shafer Theory of evidence (DS) presented below, is one attempt for such emulation.

Information fusion based on Dempster-Shafer theory of evidence

The goal is to introduce into fusion some notion of reliability of information in the individual classification streams. Unlike the classical probability where the probability of correct recognition is complementary to the probability of the error in the recognition, i.e., it assumes either correct or incorrect decision, the DS theory allows for a no decision when the evidence is not conclusive. In our work, the indicator of the reliability of

information in the individual information stream is a function of the entropy of class probability distribution in the stream. Thus, DS as applied in our work takes two vectors of posterior probabilities $(p_{a,i})$ and $(p_{b,i})$ with $i = 1, \dots, k$ from two different classifiers and produces the third probability vector $p_{c,i}$ that combines the information from both $(p_{a,i})$ and $(p_{b,i})$.

This is done as follows. First, the confidence of both MLP classifiers is computed as a nonlinear function of the entropy of the MLP outputs $\alpha = \left(1 - H/H_{\max}\right)^\gamma$ where $H = -\sum_{i=1}^k p_{a,i} \log p_{a,i}$ is the entropy of the probability distribution at the output of the MLP, and $H_{\max} = -\sum_{i=1}^k \frac{1}{k} \log \frac{1}{k}$ is the maximum possible entropy. γ introduces a nonlinearity. α is used to define the following three functions:

$$\begin{aligned} m_{a,i} &= \alpha_a p_{a,i} \\ n_{a,i} &= \alpha_a \sum_{j \neq i} p_{a,j} \\ M_{a,i} &= 1 - m_{a,i} - n_{a,i} \end{aligned}$$

where $m_{a,i}$ is the belief that the classifier a has in i -th phoneme, and $n_{a,i}$ is the belief that the classifier a has in phoneme that is other than the i -th one. In the conventional probability theory, we would have $m_{a,i} + n_{a,i} = 1$. DS theory allows for representation of the ignorance of the classifier that is given by $M_{a,i}$.

The maximal entropy of the classifier $H = H_{\max}$ indicates the situation of the maximal ignorance of the classifier. Then $M_{a,i} = 1$, $m_{a,i} = 0$, and $n_{a,i} = 0$, implying that the classifier has no information about the i -th phoneme i . On the other hand, if $H = 0$ then $M_{a,i} = 0$, implying the highest confidence in the i^{th} h phoneme.

Given beliefs $m_{a,i}$, $n_{a,i}$, $M_{a,i}$ and $m_{b,i}$, $n_{b,i}$, $M_{b,i}$ from two MLPs, they combine using the Dempster-Shafer orthogonal sum, yielding a new vector

$$m_{c,i} = \frac{m_{a,i}m_{b,i} + m_{a,i}M_{b,i} + m_{b,i}M_{a,i}}{1 - n_{a,i}m_{b,i} - n_{b,i}m_{a,i}}$$

In the extreme case when the entropy of the classifier a , $H_a = 0$, the $(m_{c,i}) = (p_{a,i})$, regardless of what the $p_{b,i}$ is. This is consistent with the product of errors rule, which states that the result is given by the classifier that yields the perfect classification, (i.e. the zero error), as indicated by $H_a = 0$.

Another extreme case arises when the entropy of the first classifier is maximal $H_a = H_{\max}$ and the entropy of the second classifier is lower than the maximum $H_b < H_{\max}$, the combined vector $(m_{c,i}) = (p_{b,i})$, conforming to the product of errors rule where the result is determined by the classifier b since the classifier a yields no useful information (indicated by $H_a < H_{\max}$).

Yet another extreme case is $H_a = H_b = 0$ but $(p_{a,i}) \neq (p_{b,i})$ (entirely reliable but conflicting information in both streams), or $H_a = H_b = H_{\max}$ (no information in either stream). In those cases, $m_{c,i} = 0$, \forall_i , i.e. the vector $(m_{c,i})$ does not have any

probabilistic interpretation, again conforming to the product of errors rule which suggest in this case the maximum probability of error.

For any intermediate values of H_a and H_b the $(m_{c,i})$ represent a weighted average of a and b , where the weights are given by the ratio α_a/α_b which is a function of H_a, H_b and the heuristic factor $\in [0, 1]$.

The values of the belief vector $(m_{c,i})$ are then used as a combined posterior vector that represents the fusion of posterior vectors $(p_{a,i})$ and $(p_{b,i})$ from the classifiers a and b . For more details of DS see (Valente and Hermansky 2006).

3.2.5.2 Combination of auditory and modulation frequencies features

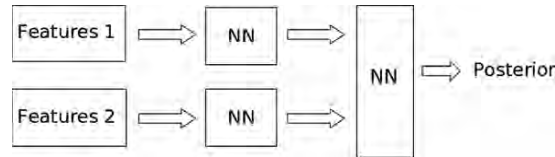


Figure 3.2: Parallel processing of two feature sets.

In conventional multi-band framework (Hermansky *et al.* 1996) a separate classifier (typically an MLP) is trained on each *auditory frequency* channel. Posterior probability estimates from individual MLPs are then merged together using another MLP in order to produce a single posterior stream estimates. Such architecture process all different *auditory frequency* channels in parallel and it is depicted in Figure 3.2. Similar processing can be applied for training a separate MLP for each *modulation frequency* channel that represent the dynamic of the speech.

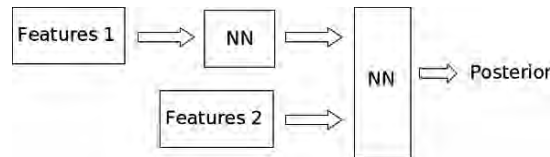


Figure 3.3: Hierarchical processing of two feature sets.

Valente and Hermansky (2007) showed that combination of classifiers trained on different frequency ranges can also be effective if performed in hierarchical (sequential) fashion.

In hierarchical processing, an MLP is trained on a first feature set to obtain phoneme posteriors. Those posteriors are then concatenated with a second feature set thus forming an input to a second phoneme posterior estimating MLP. In such a way, phoneme estimates from the first MLP are modified by a second net using evidence from a different feature stream (see Figure 3.3). In contrary to parallel processing, the order in which frequencies are presented does make a difference.

We investigate here from an ASR perspective the following issues: 1. whether there is any advantage in separate processing of modulation frequencies (similarly to separate processing of auditory frequencies); 2. whether this combination should happen in

parallel or sequential fashion. We will limit our investigation to only two frequency channels.

Let us describe the technique used to extract the different modulation channels.

3.2.5.2.1. MRASTA processing

The findings of Houtgast (1989) and Dau (1997) suggest that human perception of modulation frequencies can be modeled using a bank of filters, equally spaced on a logarithmic scale. This bank of filters subdivides the available modulation frequency range into separate channels with decreasing frequency resolution moving from slow to fast modulations.

The use of such a filterbank for ASR applications has been proposed by Hermansky and Fousek (2005) and is referred to as Multiple-RASTA (MRASTA) filterbank, as an extension of RASTA filtering. It has been shown being particularly robust to linear distortion.

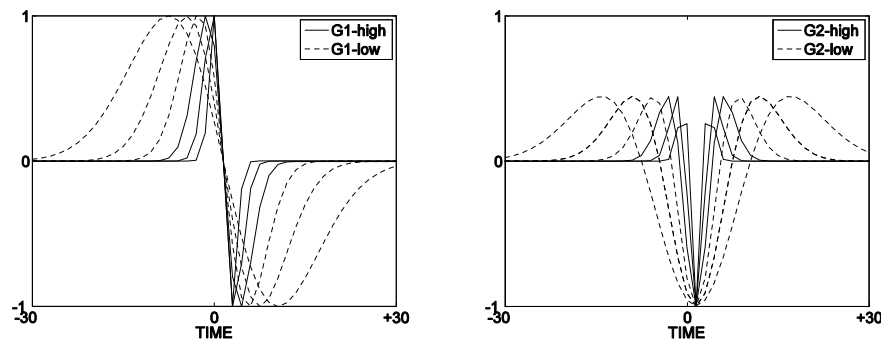


Figure 3.4: Set of temporal filter obtained by first (G1 left picture) and second (G2 right picture) order derivation of Gaussian function. G1 and G2 are successively split in two filter bank (G1-low and G2-low, dashed line) and (G1-high and G2-high continuous line) that filter respectively high and low modulation frequencies.

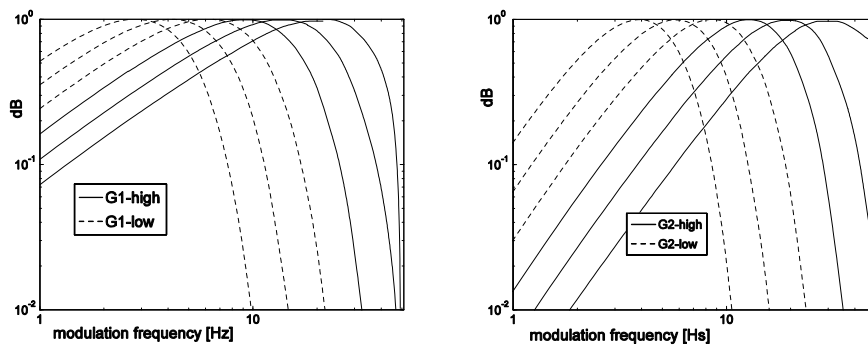


Figure 3.5: Normalized frequency response of G1 (left picture) and G2 (right picture). G1 and G2 are successively split in two filter bank. G1-low and G2-low (dashed lines) emphasize low modulation frequencies while G1-high and G2-high emphasize high modulation frequencies.

Feature extraction is composed of the following parts: critical band auditory spectrum is extracted from short time Fourier transform of a signal every 10ms. A 100ms long temporal trajectory in each critical band is filtered by a bank of bandpass filters. 15 critical bands are used. Those filters represent first derivatives (3.5) and second derivatives (3.6) of Gaussian functions with variance σ_i varying in the range 8-60 ms (see Figure 3.4). In effect, the MRASTA filters are multi-resolution bandpass filters on modulation frequency, dividing the available modulation frequency range into its individual sub-bands.⁵

$$g1_{\sigma_i} \propto -\frac{x}{\sigma_i^2} \exp\left(-\frac{x^2}{2\sigma_i^2}\right) \quad (3.5)$$

$$g2_{\sigma_i} \propto \left(\frac{x^2}{\sigma_i^4} - \frac{1}{\sigma_i^2}\right) \exp\left(-\frac{x^2}{2\sigma_i^2}\right) \quad (3.6)$$

with $\sigma_i = (0.8, 1.2, 1.8, 2.7, 4, 6)$.

In the modulation frequency domain, they correspond to a filter bank with equally spaced filters on a logarithmic scale (see Figure 3.5). Identical filters are used for all critical bands. Thus, they provide a multiple-resolution representation of the time frequency plane.

In a similar way to conventional multi-band approaches, filterbanks G1 and G2 (6 filters each) are separated in two separate filter bank G1-low, G2-low and G1-high and G2-high that filter high and low modulation frequencies respectively. We define M-high and M-low as follows:

$$\text{M-high} = [\text{M1-high}, \text{M2-high}] = [g1_{\sigma_i}, g2_{\sigma_i}] \text{ with } \sigma_i = \{0.8, 1.2, 1.8\} \quad (3.7)$$

$$\text{M-low} = [\text{M1-low}, \text{M2-low}] = [g1_{\sigma_i}, g2_{\sigma_i}] \text{ with } \sigma_i = \{2.7, 4, 6\}. \quad (3.8)$$

Filters G1-high and G2-high are short filters (Figure 3.4 continuous lines) and they process fast modulation frequencies (Figure 3.5 continuous lines). Filters G1-low and G2-low are long filters (Figure 3.4 dashed lines) and they process slow modulation frequencies (Figure 3.5 dashed lines).

The cutoff frequency for both filterbanks G-high and G-low is 10Hz. In this study 15 critical bands are used after MRASTA, thus filtering the total number of feature is $(15 \times 6) + (15 \times 6) = 180$. The fast modulations represent the original auditory spectrum with more “details” while the slow modulations give a coarse representation of the spectrum.

3.2.5.2.2. Experimental Setting

We pursue the investigation with a Large Vocabulary Continuous Speech Recognition task (LVCSR). As pilot study we use the first pass of the AMI LVCSR

⁵ Unlike those used by Hermansky (2005), filter-banks G1 and G2 are composed of six filters rather than eight, leaving out the two filters with longest impulse responses.

system for meeting transcription described by Hain (2005). The training data for this system comprises of individual headset microphone (IHM) data of four meeting corpora.

In order to use different time-frequency representation in a conventional HMM/GMM system, the TANDEM approach is used (Hermansky *et al.* 2000a). The initial time-frequency representation is used as input to an MLP which estimates phoneme posterior probabilities. MLP output are then modified using a Log/KLT transform and used as conventional features for HMM/GMM systems.

We tested a 9-frames PLP-TANDEM system (Hermansky 2000a) and the MRASTA-TANDEM system (Hermansky and Fousek 2005). In the PLP based system, 9 consecutive frames of PLP features are concatenated to form the input vector to the MLP. In the MRASTA based system, 180 outputs of the MRASTA filters (one for each filter and for each critical band) forms the input vector to the MLP.

3.2.5.2.3. Combination of Modulation Frequency Channels

We limit the splitting to only two modulation frequency ranges. FilterBanks G1 and G2 (6 filters each) are split in two separate bands as described in Section 3.2.5. Observing individual performances shows that M-high have the same overall performance as the full MRASTA filterbank. However, features obtained using M-low have noticeably worse performance.

Let us consider now the parallel and hierarchical combination schemes described in Section 3.2.5.2. Parallel processing reduces the WER by 4.4% absolute. Moving in the hierarchy from low frequencies to high frequencies yields similar performance as a single MRASTA MLP. On the other hand, moving from high to low modulation frequencies produce a significant reduction of 5.8% in final WER w.r.t. single classifier approach and of 1.4% w.r.t. the parallel combination. This is consistent with physiological experiments (Miller 2002), in which it is shown that different levels of auditory processing may attend to different rates of the modulation spectrum, the higher levels emphasizing lower modulation frequency rates.

3.2.5.2.4. Combination of Auditory Frequency Channels

Similarly to conventional multi-band approaches, the auditory spectrum is split into two frequency bands composed of lower seven and upper height critical bands respectively. MRASTA filtering is applied to those resulting in two feature set of 84 and 96 component each. We refer to those two feature set as A-Low and A-High. We combine A-Low and A-High according to the combination schemes described in Section 3.2.5.2. Parallel processing outperforms sequential processing. Using two separate frequency channels reduces the WER by 2% absolute w.r.t. the single classifier approach. Those results are consistent with findings of Allen (2005) and with conventional multi-band processing.

In conclusion, combination of classifiers trained on different ranges of auditory frequencies is more effective if performed in parallel fashion, while combination of classifiers trained on different ranges of modulation frequencies is more effective if performed in hierarchical fashion.

Results reveal that the lowest WER is obtained using the hierarchical combination of classifiers trained on separate ranges of modulation frequencies. The improvement obtained by splitting the modulation frequency channels is larger than the improvement obtained by splitting auditory frequency channels.

3.2.5.3 Applications in GALE Project

Those research results have been progressively transferred into RWTH and SRI/ICSI feature extraction systems.

	MRASTA	Hierarchical
CER	24.4%	22.3%

Table 3.9: CER on eval06 using MLP features based on MRASTA and Hierarchical.

Tests on 100 hours Chinese Mandarin recognition system with hierarchical features appended to MFCC features are reported in Table 3.9. The MLP input is augmented with log pitch estimates.

Results reveals that hierarchical features outperform conventional MRASTA by 2% absolute. In the year 3 evaluation, hierarchical features have been trained on 1500 hours of annotated speech and integrated into RWTH system for Mandarin recognition. The use of the hierarchical feature achieved a CER of 12.6% for Dev07 on the RWTH system. Details and comparison with other MLP features can be found in Section 3.2.2.

	DS	IE
CER	17.0%	17.6%

Table 3.10: CER on eval04 using MLP features combined with Dempster-Shafer and Inverse-Entropy.

Beside the hierarchical features, other combination frameworks have been proposed and applied to GALE data, in particular a multi-stream combination based on the Dempster-Shafer theory (Valente and Hermansky 2006). Dempster-Shafer (DS) combination has been compared with inverse entropy (IE) combination in the Chinese Mandarin recognition system (Hwang *et al.* 2007). Two MLP-based features (9 frames PLP-TANDEM and HATS) have been combined using DS and IE and tested in the system proposed by Hwang *et al.* (2007). Results on eval04 data are reported in Table 3.10 and show a 0.6% absolute improvement.

3.3 Improved Speech Acoustic Models

3.3.1. Introduction

Author: Katrin Kirchhoff

Recently, the predominant theme in acoustic modeling for speech recognition has been the development and refinement of novel discriminative modeling and training approaches. Traditional acoustic model training is based on the maximum likelihood criterion, i.e., model parameters (the means, variances and mixture component weights of

Gaussian mixture models) are trained to maximize the likelihood of the training data. This criterion does not directly minimize an acoustic classification accuracy score and is only indirectly related to the final system evaluation criterion (word error rate).

In line with parallel developments in the general machine learning community, attention has shifted towards discriminative approaches whose goal is to explicitly discriminate good classification hypotheses from bad hypotheses. In the context of acoustic modeling, this means that models should be trained such that the scores assigned to correct word or phone strings should be maximally distinct from those assigned to competing incorrect strings. Early work on discriminative training in speech recognition and acoustic modeling includes, e.g., (Bahl 1986; Juang and Katagiri 1992; Reichl and Ruske 1995; McDermott 1997). Recently, approaches known as minimum phone error (MPE) training (Povey 2002) and minimum phone frame error training (MFPE) (Zheng 2005), and feature-level MPE (fMPE) (Povey 2005) have become the most popular techniques. In MPE training, the model parameters are optimized to maximize the posterior probabilities of the best hypothesis for each utterance, weighted by the raw phone accuracy of that utterance. MFPE is similar in nature, except that the *frame-level* phone accuracy is used instead of the simple phone accuracy. fMPE employs a linear transformation on the acoustic features that is designed to discriminate between phone classes; it thus operates at the feature level rather than at the model level. Additional discriminative training procedures include Maximum Mutual Information (MMI) and boosted MMI (Povey 2008).

This section includes four different aspects of discriminative training. One method combines fMPE, another discriminative feature transformation (a multilayer perceptron trained on the original acoustic feature to estimate posterior phone probabilities), and MFPE within a single recognition system for Mandarin speech. The contributions from the different approaches are non-redundant, and produce additive gains. Any combination of two of the three discriminative modeling methods achieved gains larger than those obtained by each method in isolation, and the largest gain was seen when all three methods were combined. Compared to a baseline system without any discriminative training, a relative reduction in character error rate of 23.3% was achieved. In SRI's state-of-the-art GALE recognition system for Mandarin, judicious application of discriminative training techniques to different subsystems in the overall system led to improvements ranging from 20.1% to 33.3%.

The problem of unsupervised training in combination with discriminative training is addressed. Since transcribed training data is rare for certain domains, additional training data can be generated by first recognizing the untranscribed data with seed acoustic and language models, selecting the most confident automatic transcriptions according to their likelihood or confidence scores, and adding them to the pool of transcribed training data for re-estimating model parameters. This method is successful to the extent to which the automatic transcription is reliable; naturally, recognition errors diminish the gains obtained from unsupervised training. It has been observed that discriminative training is particularly sensitive to transcription errors – as shown in Yu *et al.*'s (2007) contribution; this effect is particularly severe when the untranscribed data comes from a mismatched data type, since the recognition accuracy will be lower. Instead of simple data selection, using manually transcribe a small set of the unlabeled data is proposed. This set is

selected from the pool of segments that have *low* confidence scores since these represent mismatches in data type. Results show that larger improvements on matched, as well as, mismatched data types can be obtained using such directed manual transcription instead of high-confidence data selection. The amount of additional data to be transcribed ranged around 18%, roughly equivalent to roughly 59 hours of speech.

Another method proposes the integration of a large-margin term into several existing discriminative training procedures. This renders discriminative training closer in nature to large-margin classification techniques in the machine learning community while being able to exploit conventional discriminative training setups. A direct evaluation of the importance of the margin term demonstrated that large-margin MMI training outperformed conventional MMI training on a small digit recognition task. On a large-vocabulary continuous speech recognition task, a margin-enriched MPE training procedure yielded minor improvements over standard MPE.

The final method incorporates ideas from speaker recognition into acoustic modeling for speech recognition. Two approaches are evaluated: the first applies Maximum A Posteriori (MAP) adaptation of a general background Gaussian mixture model (GMM) to each clustered phonetic state in the recognition system. The second explores a subspace representation of GMMs and resembles joint factor analysis in speaker recognition. The first approach yielded substantial improvements under maximum likelihood training, but the authors consider it unpromising for discriminative training since it is computationally expensive. The second approach makes use of fewer parameters and was evaluated on three different recognition tasks (English broadcast news, the GALE 2009 Arabic evaluation task, and the GALE 2009 Mandarin evaluation task). Modest improvements in word error rate were obtained.

3.3.2. Combining Discriminatively Trained Features, Transforms, and Models for GALE Speech-to-text Task

Authors: Jing Zheng, Xin Lei, Andreas Stolcke, Wen Wang, Mei-Yuh Hwang, Arlo Faria, Nelson Morgan, and Ozgur Cetin

3.3.2.1 Introduction

In recent years, discriminative acoustic model training techniques have led to significant accuracy improvements on many LVCSR tasks. Among these approaches, the following three categories of techniques have seen widespread use because of their proven effectiveness.

Discriminatively trained models. This category includes model parameter estimation techniques based on maximum mutual information (MMI) (Bahl 1986), minimum phone error (MPE) (Povel 2002), minimum classification error (MCE) (McDermott 1997), and other relevant criteria. Compared to traditional maximum likelihood (ML) training, discriminative training better addresses the model incorrectness problem, which is a clear theoretical limitation for the hidden Markov model (HMM)-based recognition systems. So far in LVCSR, the MPE criterion has been the most widely used, because of its

superior performance and generalization. Several modifications have been proposed to further improve MPE criterion, including the one by Zheng and Stolcke (Zheng 2005), namely minimum phone frame error (MPFE) criterion, which is employed in this section.

Discriminatively trained features. State-of-the-art LVCSR systems are mostly based on phoneme units, and therefore rely on phone discrimination. However, commonly used LVCSR features, such as Mel frequency cepstral coefficients (MFCC) and perceptual linear prediction (PLP) coefficients, are not explicitly optimized for phone discrimination. One approach that has proven effective involved multilayer perceptrons (MLPs) estimating some form of posterior phone probabilities at the frame level, to be used as HMM observation features. This approach is known as Tandem acoustic modeling (Ellis 2001). MLP features are then typically used in combination with regular features, such as MFCC and PLP, to obtain maximum benefit. The resulting combined feature typically has high dimensionality.

Discriminatively trained transforms. The idea of these approaches is to use discriminative criteria to estimate linear feature transforms, which make corrections to standard features to improve discriminative power. Typical approaches include fMPE (Povey 2005) and MPE-RDLT (region dependent linear transforms) (Zhang 2006). Compared to MLP features, which are usually appended to regular features, these approaches modify the standard features themselves through transformation based on certain conditions.

These three categories of discriminative training methods attack the speech modeling problem at different levels. Can they work together? Are their contributions non-redundant? In the remainder of the section, we answer this question by applying MLP features, fMPE transforms, and MPE training to a large vocabulary Mandarin recognition task, and by studying how the methods interact.

3.3.2.2 Combined Discriminative Modeling

We first review the three techniques we used in this study: MLP features, MPE models, and fMPE transforms. Then we propose the combined approach and compare results between different configurations.

3.3.2.2.1. MLP Features

We built our system following the Tandem approach (Ellis 2001), i.e., features generated by the connectionist (MLP) system were used as HMM observation data. This approach produced significant word error rate reduction, about 10% relatively in an English conversation telephone speech (CTS) system (Zhu 2005), and was shown to generalize across domains and even languages, making it even more attractive (Stolcke 2006).

We used two types of MLPs to generate features. A PLP/MLP, which focuses on medium-term information, was trained on 9 consecutive frames of PLP features, as well as, their first and second order differences. Hidden activation temporal pattern-MLPs (HATs) (Zhu 2004; Morgan 2004), which focus on long-term information, extracted

information from 500 ms windows of critical band energies. Both PLP/MLP and HATs systems generated phone posteriors, which were combined using inverse-entropy weighting (Misra 2003) and then projected to 32-dimensional features via a **Karhunen-Loeve** transform.

We appended the 32-d MLP features with the 39-d MFCC features and the 3-d pitch features (consisting of log pitch and its first and second order differences) features to form a 74-dimensional feature vector, which was then used to train the HMM system.

3.3.2.2.2. MPFE Models

MPE model training was first proposed by Povey (2002). Gaussian parameters are estimated to optimize the following objective function:

$$F_{MPE}(\lambda) = \sum_{r=1}^R \sum_s P_k(s|O_r, \lambda) \text{RawPhoneAccuracy}(ies) \quad (3.9)$$

where $P_k(s|O_r, \lambda)$ is the posterior probability of hypothesis s for utterance r given observation O_r , current parameter set λ , and acoustic scaling factor k . $\text{RawPhoneAccuracy}(ies)$ is a measure of the number of correctly transcribed phones in hypothesis s . So the MPE objective function is the weighted average phone accuracy in the lattices generated by a LVCSR system.

Optimization of the MPE criterion employs an adapted extended Baum-Welch algorithm, which was initially used for MMI training. MPE training aims to minimize phone error rates, which in turn leads to lower word error rates. MPE training is shown to produce more accurate models than MMI training, and also appears to have superior generalization from training to test data, compared with direct word error rate minimization (MWE) (Povey 2002), and therefore, has been adopted widely.

In this study, we used a variant of MPE training, optimizing the weighted average of frame-level phone accuracies of phone lattices generated from a fast decoding system (Zheng 2005). We have shown that this approach, also known as MPFE, had comparable or better results than the standard MPE in English and Mandarin CTS system, when combined with a MMI model prior and I -smoothing (Zheng 2005). This observation has also been confirmed by other researchers (Povey 2007).

3.3.2.2.3. fMPE Transforms

fMPE Transforms were first proposed by Povey (2005). The idea is to use a very large linear transform that projects a high-dimensional feature vector into the standard feature space, as a correction to the original feature to optimize the MPE objective function of Equation (3.9)

$$y_t = x_t + Mh_t \quad (3.10)$$

where x_t is the original feature vector at time t ; h_t is a high dimensional feature vector; M is the transform matrix that needs to be estimated, and y_t is the corrected feature that has improved discriminative power according to the MPE criterion. fMPE optimization uses first-order gradient descent, with iterative transform estimation and ML model

updates. It has been shown that subsequent MPE training after the fMPE transform is fixed can achieve additional word error rate reductions.

Similar to (Povey 2005), we constructed h_t by computing Gaussian posteriors from evaluating a Gaussian mixture model (GMM), and with neighboring context expansion, which combines posterior vectors of neighboring frames into an extended vector. The GMM used for this purpose had 102K Gaussians trained on MFCC+pitch features (Lei 2006) with 42 dimensions. We also used a second layer of 3210 Gaussians clustered from the 102K Gaussians for fast computation.

3.3.2.2.4. The Combined Approach

Although both MLP features and fMPE transforms aim at improving the discriminative power of ASR input, they have very different implementations and internal principles, which leave room for effective combination. In this study, we used the 74-d MFCC+pitch+MLP features as input, a 102K dimensional posterior vector with context width 3, resulting in an fMPE transform M of size $74 \times 3 \times 102K$. We typically ran 3-4 iterations of fMPE optimization. Then, with the fMPE transform fixed, we ran 4-5 iterations of MPFE model updates. Thus, the system would incorporate all three discriminative training approaches.

To have a comparison with the combined approach, we also trained fMPE transforms without the MLP features (i.e., with the 42-d MFCC+pitch features only). In this setup, because of the lower input feature dimensionality, we increased the context width to 5 to have an fMPE transform of size closer to the previous one: $42 \times 5 \times 102K$. After the fMPE transform was learned, we ran MPFE training. We also tested the other 6 possible combinations, such as the MLP features with MPFE training but no fMPE transforms, etc. We first ran a comparison with a speaker-independent (SI) single pass bigram system built on a within word triphone acoustic model and MFCC+pitch frontend processing. The acoustic model was trained on 465 hours of speech, including LDC Mandarin Hub4, Mandarin TDT4 closed captions filtered by flexible alignment (Venkataraman 2004), and contained about 200K Gaussians. The bigram was highly pruned for fast decoding. Testing was performed on the dev06bn test set, which consists of 3.5 hours of Mandarin broadcast news.

Sysid	MLP	MPFE	fMPE	WER	Rel.
S0				17.1%	-
S1	X			15.3%	-10.5%
S2		X		14.6%	-14.6%
S3			X	15.6%	-8.8%
S4	X	X		13.4%	-21.6%
S5	X		X	14.7%	-14.0%
S6		X	X	13.9%	-18.7%
S7	X	X	X	13.1%	-23.3%

Table 3.11: CERs on dev06bn test set with 1-pass speaker-independent bigram decoding.

As Table 3.11 shows, the contributions from the three discriminative training approaches were nonredundant, and the fully combined approach (S7) worked the best among all configurations. Compared to the baseline S0, which used none of the three

discriminative training approaches, the CER reduction was quite significant: 4.0% absolute or 23.3% relative.

3.3.2.3 Applying to GALE STT System

To maximize accuracy performance, most state-of-the-art LVCSR systems employ multi-pass decoding architecture involving several sub-systems integrated with cross-adaptation and system combination. Subsystems are typically designed to be different from each other but with similar performance in order to maximize the combination effect. In SRI's latest GALE 2008 STT system, three subsystems using different acoustic and language models were employed, as shown in Figure 3.6:

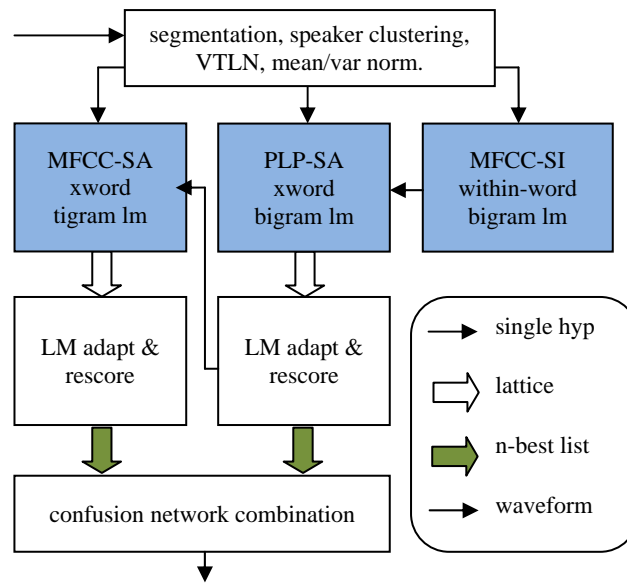


Figure 3.6: System decoding architecture

- A. MFCC-SI: This system was built on a within word triphone model using an MFCC+pitch frontend. A pruned bigram language model was used for fast SI decoding to generate adaptation hypotheses for later passes.
- B. PLP-SA: This system employed a crossword triphone model trained with a PLP+pitch frontend and a pruned bigram language model. The model was first adapted to the hypotheses generated by MFCC-SI system using CMLLR-based speaker adaptive training (SAT) and MLLR adaptation, and then performed a speaker-adapted (SA) decoding to generate lattices, which was rescored with adapted language models to generate N-Best lists (See Section 3.5.4).
- C. MFCC-SA: Similar to PLP-SA except that this system was trained with MFCC+pitch frontend, and was adapted to hypotheses from PLP-SA decoding.

The N-Best lists of this system were combined with the one from PLP-SA using confusion network combination to generate final output.

All the acoustic models were trained on 1642 hours of speech data from Hub4 and GALE program releases. Language models were developed from corpora comprising 1.4 billion words from various sources. More details can be found in Section 3.5.4.

Since system combination favors diversified and close quality subsystems, we applied MLP feature and MPFE training to MFCC-SI and MFCC-SA, but fMPE and MPFE to PLP-SA. This design makes subsystems diversified and balanced, and saves significant amount of training and testing time compared with applying the all three discriminative training techniques to all the acoustic models. Based on the results of some internal experiments, the performance difference between the two approaches was minimal.

We tested this system on two different test sets: GALE 2007 development set (dev07), which contains 2.4 hours of speech data from 74 shows, with 1.0 hour broadcast news (BN) and 1.4 hour broadcast conversation (BC); and GALE 2008 development set (dev08), which has one hour of speech from 44 shows, with 0.5 hours each in BN and BC. For comparison, we also ran a system that shared the same search structure but without using any of the discriminative training approaches, as the baseline. The CER results of these two configurations are compiled in Table 3.12.

	DEV07			DEV08		
	bn	Bc	all	bn	bc	all
Baseline (ML)	3.3	18.7	12.1	3.9	16.9	10.6
Disc. training	2.3	14.2	9.1	2.3	13.5	8.3
-Rel. Δ (%)	30.3	24.0	24.7	33.3	20.1	21.7

Table 3.12: CER% of the full system on dev07 and dev08.

As Table 3.12 shows, in all cases the system with discriminative training performed significantly better than the baseline, ranging from 20.1% to 33.3% CER reduction. Compared with the simple single-pass system experiment, the gain from discriminative training did not diminish because of the use of much larger language model and adaptation techniques. The fact that the improvements are similar in two different test sets also shows the discriminative training techniques generalize well.

3.3.2.4 Conclusions

We have presented a discriminative training approach combining MLP features, fMPE transforms, and MPFE training. Experimental results showed that the contributions from these three individual approaches were nonredundant. Applying the combined approach in a state-of-the-art Mandarin GALE STT system led to significant character error rate reductions on two different test sets with both broadcast news and broadcast conversations data, ranging from 20.1% to 33.3%.

3.3.3. Unsupervised Training and Directed Manual Transcriptions for Mandarin Speech-to-Text

Authors: Kai Yu, Mark J. F. Gales, Lan Wang and Phillip C. Woodland

3.3.3.1 Introduction

A recent trend in building large vocabulary speech recognition system is to use very large training sets for parameter estimation of the acoustic models. For some tasks, such as automatic transcription of Broadcast data, it is fairly easy to obtain thousands of hours of audio data from radio and television shows. However, in order to train the acoustic models, word level transcriptions are required. Hence, a major cost of using large amounts of broadcast data is the provision of accurate manual transcriptions. Approximate manual transcriptions, such as closed captions, can be used with *lightly-supervised training* (Lamel *et al.* 2002; Chan and Woodland 2004). This leads to low error rate semi-automatic transcriptions which yield good performance for hidden Markov model (HMM) parameter estimation using both maximum likelihood (ML) (Lamel *et al.* 2002) and discriminative criteria (Chan and Woodland 2004). However, in some cases, for instance broadcast news transcription in Arabic and Mandarin, even approximate transcriptions are not available. In these scenarios, *unsupervised training* techniques must be applied, where a seed model is used to recognize the unsupervised audio and the automatic transcriptions are used during training. Most previous studies of unsupervised training have concentrated on ML estimation techniques (Lamel *et al.* 2002; Kamm and Meyer 2002; Wessel and Ney 2005; Zavalagkos and Colthurst 1998). However, the majority of state-of-the-art speech recognition systems make use of discriminative training approaches such as Minimum Phone Error (MPE) (Povey and Woodland 2002). This section will describe an in-depth study of how discriminative training can be applied to an unsupervised training task and how the various problems that arise can be addressed.

Discriminative training criteria are more sensitive to the accuracy of the transcriptions than ML training. This is because discriminative criteria aim to explicitly increase the “distance” between the “reference” and all other possible word sequences. The quality of the “reference” and the “distance” could be two problems during discriminative training. First, transcription errors may limit performance gains especially when the nature of the training data is not well matched to that of the seed model. Second, the acoustic model used to generate the reference hypothesis is the same as, or similar to, the one to be trained. Thus, the ASR derived reference hypothesis tends to be the most probable word sequence. This results in better than expected discriminative criteria values, which can limit possible discriminative training gains. Data selection can help to address the first problem, by identifying regions where the transcriptions are felt to be reliable. The second problem can be partly addressed by simplifying the language models used during discriminative training compared to the one used to obtain the hypotheses.

Fully unsupervised approaches, even using the above schemes, are shown to yield only limited gains with discriminative training compared to fully manually transcribed data. This effect is most noticeable where the seed acoustic and language models are not

closely matched to the data (Wang *et al.* 2007). An approach based on active learning (Cohn *et al.* 1994) will then be discussed, *directed manual transcription*. Here subsets of the training data are automatically identified to be manually transcribed. By appropriately selecting the “best” subset to transcribe, rather than using an ad hoc selection scheme, the optimal balance of transcription cost for the maximum reduction in word error rate may be achieved (Yu *et al.* 2007).

3.3.3.2 Unsupervised Discriminative Training

The previous section has given an overall introduction of unsupervised training. This section will investigate in detail the issues of unsupervised discriminative training.

3.3.3.2.1. Training Procedure

The general procedure of unsupervised training is to first recognize the unsupervised audio using the seed acoustic and language models. Data selection approaches may be used to filter out poorly recognized data. These data are often discarded. The unsupervised audio with the automatic transcriptions are then folded into the original training set for acoustic (ML and discriminative) and/or language model training.

In this work, MPE criterion (Povey and Woodland 2002) was used to train discriminative systems. During the decoding of unsupervised audio, a multi-pass decoding system using *adapted* MPE models is used. The adapted decoding system utilizes a similar procedure to (Sinha *et al.* 2006). This is the lattice generation stage in the state-of-the-art Cambridge multi-pass recognition framework. In this work, the Viterbi 1-best recognition result of the adapted decoding system is used as the output transcription, as this is felt to give a good balance between deletions and insertions. To get confidence scores for data selection, confusion network (CN) decoding (Evermann and Woodland 2000) is also performed on the output lattices, yielding another set of transcriptions with associated confidence scores. These transcriptions are only used in the data selection stage rather than for reference generation, due to the larger deletion rate.

The first step of data selection on the Mandarin Chinese broadcast task is to remove the non-Mandarin shows (Wang *et al.* 2007). Further data selection is then performed to filter out the poorly transcribed unsupervised data (Wang *et al.* 2007; Ma *et al.* 2006). This selection can be done at segment-level or show-level, both are based on the word confidence scores generated from the adapted decoding system. The show or segment level confidence scores are calculated by averaging the confidence scores of each word:

$$C_S = \frac{\sum_{W \in S} C_W T_W}{\sum_{W \in S} T_W} \quad (3.11)$$

where S can be one show or one segment, C_S is the averaged confidence score of S , C_W is the confidence score of word W within S calculated by the multi-pass decoding system, T_W is the duration of word W . A threshold on C_S is then set to split the unsupervised data into two parts. The traditional selection strategy is to simply remove those segments that fall below a set threshold (Ma *et al.* 2006; Wang *et al.* 2007).

It is worth noting that the above confidence score based data selection implicitly performs data type classification as the seed model will normally recognize the

unsupervised data with matched type better than those data which are mismatched. Consequently the confidence scores are higher for the matched data types. For example, in this work, the Mandarin broadcast task contains two types, broadcast news (BN) and broadcast conversation (BC). Experiments have shown that the two types of data have significant differences in both the acoustic and transcription statistics (Wang *et al.* 2007). As the seed models are trained on supervised BN data, the unsupervised BC data will have lower confidence scores. Figure 3.7 shows the segment-level confidence score distribution of BN and BC data in this work. Two peaks corresponding to BC and BN can be observed. As the BC/BN type is defined at the show level, the distinction at segment-level is smoothed.

After the data selection stage, a certain amount of unsupervised audio is obtained, with the corresponding transcriptions, which is treated as if they are the correct manual transcriptions. These data are then folded into the original training set to form a larger training set. The acoustic models are then updated on the newly formed training set as usual. Initially, maximum likelihood (ML) training is performed, which is then followed by MPE training.

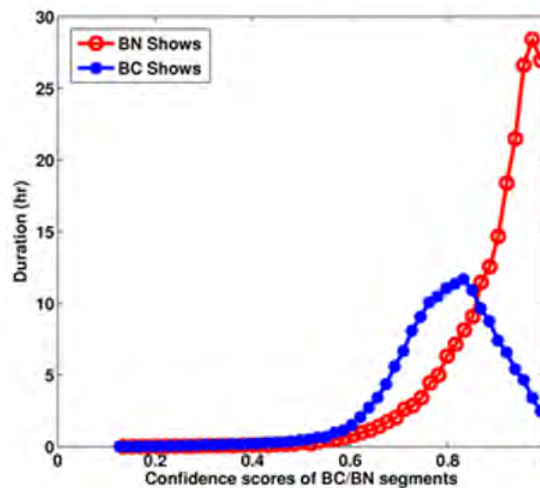


Figure 3.7: Segment-level confidence score distribution of BC and BN data

3.3.3.2.2. Data Summary and Experimental Setup

Sys.	Data	Trans.	Size (hour)		
			BN	BC	ALL
S0	baseline	manual	166.7	19.7	186.4
S1	+ subset1	manual	363.2	150.8	514.0
S2	+ subset1	auto	352.3	152.2	504.5

Table 3.13: Data summary

In this work, the baseline seed model, referred to as **S0**, was initially trained on manually transcribed data. An additional dataset, referred to as **subset1**, with manual transcription was used for unsupervised training experiments. This allows a contrast of

the use of unsupervised training with supervised training.⁶ **S1** is the model trained on the combined data set of the **S0** training data and **subset1** data set with manual transcription. **S2** is the comparable unsupervised training system to **S1**. The data is summarized in Table 3.13.

Two test sets were used to evaluate the different systems: **bnmdev06** and **bcmdev05**. **bnmdev06** comprises 3.6 hours of data taken from a range of BN sources, referred to as the BN testset later. It includes some of the standard existing test sets described by Sinha *et al.* (2006), dev04f (0.5 hour), eval03m (0.6 hour) and eval04 (1.0 hour). In addition, a more recent set of 4 shows (1.6 hour) taken from July - October 2006 were also used. The bcmdev05, BC test-set, comprises 2.5 hours of data taken from 5 BC shows during March 2005.

In all acoustic systems, base features were 13 PLP Cepstral coefficients with their 1st to 3rd derivatives. Cepstral Mean Normalisation (CMN) was applied and HLDA was used to project the dimension down to 39. Pitch was then extracted, and appended to the features along with its 1st and 2nd order derivatives (Gales *et al.* 2005). State-clustered triphone HMMs, with 6K distinct states and an average of 36 Gaussian components per state were used for all systems. The baseline language models were word-based interpolated tri-gram and 4-gram models, trained on about 1.2G words with a 51K multi-character word list (Wang *et al.* 2007). The baseline acoustic and language models were used in recognizing the unsupervised audio within an adapted decoding system. In contrast, in all experiments on the test datasets unadapted single pass Viterbi decoding using the tri-gram baseline LM was used unless explicitly stated.

3.3.3.2.3. Unsupervised Training Issues

Sys.	BN		BC	
	ML	MPE	ML	MPE
S0	15.1	13.6	29.3	25.4
S1	12.8	10.5	26.1	21.8
S2	13.8	12.0	27.7	24.8

Table 3.14: Unadapted single pass performance of supervised training and unsupervised training

The initial performance comparison between supervised and unsupervised training is shown in Table 3.14. From this table, it can be seen that, with ML training, using only automatic transcriptions (**S2**) leads to a similar absolute reduction in CERs for both BN and BC data: 1.3% for BN and 1.6% for BC compared to the **S0** performance. In contrast, for MPE training, the gain on BC is only 0.6%, which is less than half of that on BN (1.6%). This demonstrates that the higher error rate for BC data can significantly affect the performance of unsupervised discriminative training. The second row of Table 3.14 is the performance of supervised training, which gives an upper bound of all possible gains from unsupervised training. For MPE training, the CER reduction of supervised training

⁶ It is worth noting that the actual amount of data used for unsupervised training (before confidence score based data selection) is slightly less due to automatic segmentation and non-Mandarin show removal.

compared to **S0** is 3.1% on BN and 3.6% on BC. For the **S2** MPE system, the proportion of the CER reduction from supervised training is 57% on BN, while on BC only 17%. This shows that for unsupervised training, if the type of the test data matches the well recognized unsupervised data, the performance gain is closer to the gain by supervised training, otherwise, it may be greatly limited.

Data selection may be performed to address the problem. Segment-level selection was used in this work unless explicitly stated. The traditional unsupervised discriminative training approach is to discard the data with confidence scores falling below a threshold and use the rest for training. It is interesting to investigate the impact of the thresholds on the unsupervised training data selection performance. Four thresholds 0.7, 0.77, 0.8 and 0.86 were used. Figure 3.8 shows the relative reduction in CER of unsupervised training with automatic transcriptions, versus using supervised training on all of this data. The x-axis shows the percentage of the data that remained after data selection. The y-axis shows the percentage of the unsupervised training gain from supervised training. The relative improvement of discriminative training corresponding to the confidence score threshold C , G_C is calculated using

$$G_C = 100 \frac{\mathcal{E}_C - \mathcal{E}_{S_0}}{\mathcal{E}_{S_1} - \mathcal{E}_{S_0}} \quad (3.12)$$

where \mathcal{E}_C is the CER of the model trained on unsupervised data selected using confidence score threshold C . \mathcal{E}_{S_1} is the CER of supervised training and \mathcal{E}_{S_0} is the performance of the baseline **S0** model, which are shown in Table 3.14.

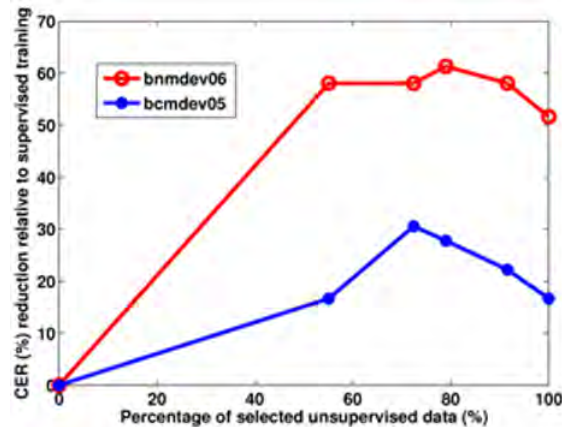


Figure 3.8: Reduction in CER (%) from unsupervised MPE training with automatic transcriptions relative to that from supervised training

From Figure 3.8, the relative improvement did not always increase as more data are selected. This is expected because if too little data are selected, they will contribute less to model training; if too much data are selected, the quality of some of the selected data is poor. The threshold should be selected to balance the two issues. Furthermore, it can be observed that the relative improvement on BN is always greater (about twice) than the gain on BC. There was always a large amount of BN data selected when confidence score

based selection was used. Hence, the poor MPE gain on BC is also expected. The figure again illustrates that the performance improvement on the data set with mismatched type is limited for discriminative training. Experiments using more unsupervised data did not yield further improvement either (Wang *et al.* 2007). This shows that the use of only automatic transcription does not yield sustainable improvements on the test data with mismatched type.

3.3.3.3 Directed Manual Transcriptions

An alternative strategy to the pure use of automatic transcription is to manually transcribe a small amount of data to improve the acoustic model training. The data with manual transcriptions are then used together with the other automatically transcribed data for acoustic model training. In order to improve the recognition performance on the data with mismatched type, the data with low confidence scores were selected for manual transcription. This is referred to as *directed manual transcription* (Yu *et al.* 2007). Though this method requires some manual effort, it is less costly than manually transcribing the complete dataset. The aim is to find a reasonable trade-off between the increase in cost and the recognition performance.

Sys.	S0 + (hr)		CER (%)	
	Man.	Auto.	BN	BC
S0		0	13.6	25.4
S2	0	318.1	12.0	24.8
—	0	251.3	11.7	24.4
—	58.9	0	12.7	23.4
S2c	58.9	251.3	11.4	23.3
S1	327.6	0	10.5	21.8

Table 3.15: Unadapted single pass CER (%) of unsupervised MPE systems trained on subset1 data with different amounts of manually or automatically transcribed data

Table 3.15 shows the performance of unsupervised MPE training with different data selection strategies, which vary the amount of data manually transcribed using confidence score based selection.⁷ From this table, the use of complete manual transcription (S1) yielded a 3.1% absolute reduction in CER on BN and a 3.6% absolute gain on BC, compared to the baseline S0 system. Using only automatic transcriptions, a 1.6% CER reduction on BN and 0.6% on BC was obtained. This is 52% relative for BN and only 17% on BC with respect to the supervised training CER reduction. From the third row of table 3, performing confidence score based data selection increases the gain proportion to 61% on BN and 28% on BC. Though there are improvements on both test sets, the relative improvement on BC is still poor. The fifth row shows the performance of directed transcriptions together with automatic transcriptions, which is referred to as S2c. By transcribing about 18% of the unsupervised data, the S2c system obtains a

⁷ The confidence score threshold used was 0.77, which has been shown as a good trade-off in selecting pure automatic transcriptions.

significantly larger reduction in CER on BC (2.1%), which is now comparable to the improvement on BN (2.2%). The percentage of the supervised training improvement is 71% on BN and 58% on BC. Therefore, manually transcribing a small amount of poorly recognized data can double the relative BC improvements without reducing the performance on BN. The fourth row shows the experiment where only the data with directed manual transcription was added, i.e., no automatic transcriptions were used. Comparing the fourth and fifth rows, purely folding in the data with manual transcription yielded significantly poorer performance on BN while better performance on BC. This is expected because the data with manual transcription are mostly BC data, while the data with automatic transcription are mostly BN data, hence, the model set is heavily tuned to BC data. It is also clear that adding the automatic transcription is beneficial for the BN data, while for the BC data, it is just slightly better than the use of only manual transcriptions. This, again, illustrates the issue of unsupervised discriminative training with fairly high error rate transcriptions.

3.3.3.3.1. Confidence Score Threshold Selection

As in the previous section, four confidence score thresholds were used. Experiments showed that the BC type of data were always dominant in the selected data set for all thresholds. This is consistent with Figure 3.7 as BC data normally have lower confidence scores. This implicit BC/BN selection may contribute to the improved gain on BC test set.

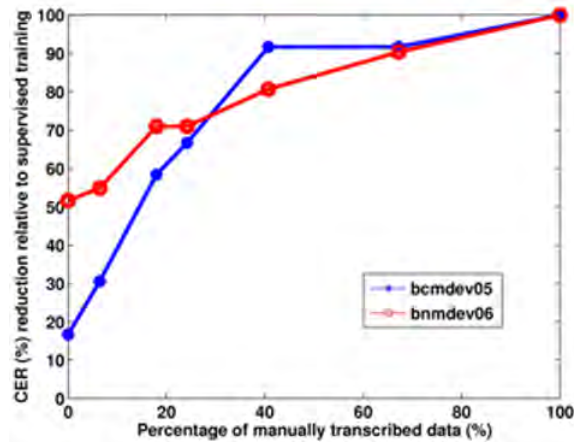


Figure 3.9: Reduction in CER (%) from unsupervised MPE training with directed manual transcriptions relative to that from supervised training

Figure 3.9 gives the proportion of the MPE training improvements when adding varying amount of manual transcriptions. It can be seen that the reductions in CERs increase as more data is added at the cost of producing the additional manual transcriptions. The increase on BC is faster than on BN when transcribing a small amount of data. Transcribing less than 20% of the data can yield over 50% of the complete supervised training improvement for BC data because the data selection process tends to select BC data.

3.3.3.3.2. Comparison to Random Selection

The previous sections have shown the advantage of adding manual transcriptions for lower confidence score segments during unsupervised training. It is also interesting to investigate whether this selection approach is preferable to random selection.

Table 3.16 gives the comparison between confidence score based directed manual selection and random selection. The first row in the table is the random selection from all unsupervised data, which has similar amount of data as the S2c system in Table 3.15. It is clear that confidence score based selection outperforms random selection on both test sets for MPE training. In particular, the improvement on BC over random selection is larger than on BN since directed selection favors BC data. The second row is the random selection on only the BC part of the unsupervised data, which ensures only BC data is selected. The confidence score based selection still outperforms the random selection on BC data on both test sets for MPE training, though the improvements are smaller. This illustrates that selecting lower confidence segments is preferable to selecting segments of a specific type during unsupervised discriminative training. This further demonstrates the essential issue in unsupervised training is the high error rate transcriptions.

Sys.	S0 + (hr)		CER (%)	
	Man.	Auto.	BN	BC
Rand. (ALL)	58.9	248.7	11.7	23.9
Rand. (BC Only)	58.9	254.6	11.5	23.6
S2c	58.9	251.3	11.4	23.3

Table 3.16: Unadapted single-pass CER (%) of MPE trained system trained on **subset1** data with directed manual selection and random selection

3.3.3.4 System Refinement

The previous sections have described the basic systems using directed manual transcriptions. This section will discuss some refinements.

One refinement is to use the additional transcriptions for language model estimation as well. The basic procedure is to build separate language model components for the added transcriptions and then interpolate among the newly estimated components and the old language model components (Wang *et al.* 2007). However, it was found that, with segment-level data selection, rebuilding language model did not give additional gains on any of the test sets. Therefore, further discussion will only concentrate on acoustic model training.

3.3.3.4.1. Multi-Pass Adaptation System

All previous experiments were using unadapted single pass decoding. It is also interesting to know how unsupervised training performs after adaptation. The speaker adapted decoding system as in Section 3.3.3, was used to test the adaptation performance.

Sys.	S0 + (hr)		CER (%)	
	Man.	Auto.	BN	BC
S0	0	0	11.7	24.2
S2	0	318.1	10.8	23.3
S2c	58.9	251.3	10.3	22.0
S1	327.6	0	9.4	20.4

Table 3.17: Multi-pass adaptation system performance of unsupervised training and supervised training

From Table 3.17, after adaptation, the supervised training **S1** system yields a 2.3% absolute reduction in CER on BN and 3.8% on BC compared to the baseline **S0** system. Using complete automatic transcription, the **S2** system obtains 39% of the supervised training improvement on BN, while only 24% on BC. With 18% data manually transcribed, the **S2c** system yields 60% of the gain on BN and 59% on BC. The relative improvement on the BC data is even larger than the relative improvement in unadapted single pass decoding. This further illustrates the effectiveness of using directed manual transcription.

3.3.3.4.2. Show-Level Data Selection

The previous experiments are all based on segment-level confidence scores because the segment based selection is felt to be finer than show-level selection. However, in practice, it is not always convenient to manually transcribe non-contiguous segments, and hence often manual transcriptions are produced at show level. Therefore, it is interesting to compare the previous results with directed manual transcription at the show level.

Conf Level	S0 + (hr)		Unadapted		Adapted	
	Man.	Auto.	BN	BC	BN	BC
show	58.9	257.9	11.7	23.6	10.5	22.1
seg	58.9	251.3	11.4	23.3	10.3	22.0

Table 3.18: Comparison between show-level and segment-level directed data selections

Table 3.18 gives the comparison between show-level and segment-level selection of directed manual transcriptions. The “unadapted” column was unadapted Viterbi decoding, while the “adapted” column was using the multi-pass adaptation system for decoding. For the show level selection, a threshold of 0.80 was selected to yield similar amount of data for manual transcription as the **S2c** system (the second row). With a similar amount of manual transcription (58.9 hours), segment level selection outperforms show level selection on both test sets for MPE training with and without adaptation. Comparing show level selection to the baseline **S0** performance in Table 3.17 and Table 3.15, for unadapted decoding performance, the proportions of the supervised training improvement are 61% for **BN** and 50% for **BC**; for adaptation performance, the proportions are 52% for **BN** and 55% for **BC**. Those relative improvements are still good and just slightly smaller than the segment level selection. This illustrates that show level selection can be efficiently used in directed manual transcription.

3.3.3.5 Conclusion

Sensitivity to transcription errors is a significant issue in unsupervised discriminative training for LVCSR. Data selection is normally used to filter out poorly transcribed data and the high quality automatic transcriptions of unsupervised data are used for discriminative training. However, this approach does not yield good performance on data of a different genre from the training data of the seed model. To address this, we have described an alternative approach based on directed manual transcription. Here, a small amount of high error rate data are manually transcribed to supplement the automatic transcription. The performance of unsupervised MPE training were evaluated on a Mandarin transcription task, where both Broadcast Conversation (BC) and Broadcast News (BN) data were used. Experiments showed that incorporating directed manual transcription can effectively improve the discriminatively trained system on the BC data compared to the traditional unsupervised approach in both unadapted and adapted decoding. It was shown that confidence score based data selection outperforms random data selection. Though segment-level confidence scores yielded better performance, show-level confidence scores can also lead to reasonable improvements and are easier to use in practice.

3.3.4. Integration of Large Margin Concept into Standard Discriminative Training Criteria

Authors: Georg Heigold, Thomas Deselaers, Ralf. Schlüter, Hermann Ney, George Saon, and Daniel Povey

3.3.4.1 Introduction

The task of parameter estimation includes two competing terms. The loss function accounts for the systems' risk (e.g., phoneme (Povey 2002)) while another term controls the model complexity (e.g., I-smoothing (Povey 2003) or margin (Vapnik 1995)). Depending on the training conditions, we expect different relative importance of these two terms, see Table 3.19.

Loss	vs.	Margin
infinite data	↔	sparse data
many training errors	↔	few training errors

Table 3.19: Relative importance of loss and margin term under different training conditions.

In this work, we would like to find out the potential of a margin term in Automatic Speech Recognition (ASR). More explicitly, our objectives for this work are:

1. Direct evaluation of the margin term. Ideally, we can turn on/off the margin term in the optimization problem. In particular, we want to avoid effects arising from different loss functions, optimization algorithms, model parameterizations, convergence speed etc. Unfortunately, but similar to (most) other approaches, we cannot exclude the effect of spurious local optima.

2. Evaluation on a state-of-the-art LVCSR system. Ideally, we directly improve on the best discriminative criterion, e.g., MPE.
3. Showing a clear relationship to existing large margin classifiers.

For this purpose, we transform the standard large margin optimization problems including constraints and slack variables into equivalent optimization problems that resemble conventional training criteria, e.g., MMI or MPE. Like for Minimum Classification Error (MCE), the loss function is replaced with a smooth approximation (Jebara 2002; Zhang 2003). The practical advantage of this approach is that the incorporation of the margin term leads only to small modifications of the conventional training criteria. In particular, this means that the same efficient algorithms can be used with basically the same implementation. Keep in mind that these are modifications of the training criteria and do not affect the underlying model.

3.3.4.2 Common Large Margin Classifiers

We shall use two different formulations of the large margin optimization problem found in the literature. We directly introduce these optimization problems in the form that best fits our purpose and refer the reader to the literature for the original formulation. Then, we replace the non-smooth loss function by a smooth approximation such that we can use standard gradient-based algorithms for optimization, e.g., Extended Baum Welch (EBW) or Rprop.

3.3.4.2.1. Support Vector Machines (SVMs)

According to (Altun 2003), for C classes and N labeled observations ($x_n \in \mathbb{R}^D, c_n \in \mathbb{N}$), the optimization problem of SVMs w.r.t. the model parameters $\Lambda = (\lambda_1 \in \mathbb{R}^D, \dots, \lambda_c \in \mathbb{R}^D)$ can be formulated as follows -

$$\hat{\Lambda} = \operatorname{argmin}_{\Lambda} \left\{ \frac{1}{2} \|\Lambda\|^2 + \frac{J}{N} \sum_{n=1}^N l(c_n, d_n; 1) \right\}. \quad (3.13)$$

For SVMs, the distance vector d_n has the components $d_{nc} := (\lambda_{c_n} - \lambda_c)^T x_n$. The empirical constant $J > 0$ is used to balance the L2-norm regularization $\|\Lambda\|^2 := \sum_c \|\lambda_c\|^2$, and the loss term. The hinge loss function is the typical loss function in the context of SVMs. The multiclass hinge loss function is defined as -

$$l^{(hinge)}(c_n, d_n; \rho) := \max_{c \neq c_n} \{-d_{nc} + \rho, 0\} \quad (3.14)$$

In this formulation, ρ is kept fixed and the model parameters Λ are scaled to adjust the effective margin. The optimization problem in Equation (3.13) tries to simultaneously maximize the margin and minimize the hinge loss.

3.3.4.2.2. Large Margin Classifier

Similar to (Li 2007), an alternative formulation uses the margin parameter ρ directly in the objective function (Saon 2008) –

$$\hat{\Lambda} = \operatorname{argmin}_{\Lambda} \min_{\rho} \left\{ -\rho + \frac{1}{N} \sum_{n=1}^N l(c_n, d_n; \rho) \right\}. \quad (3.15)$$

The notion of the distance is generalized to $d_{nc} := \log p_{\Lambda}(c_n, x_n) - \log p_{\Lambda}(c, x_n)$ in the usual way. Again, the typical loss function is the hinge loss function, $l^{(hinge)}$. In addition to the model parameters Λ , the parameter ρ is optimized in Equation (3.15). For the experiments reported below, the parameter was tuned beforehand on some development data and then kept fixed, both for training and recognition. This formulation is similar to the SVM formulation in Section 3.3.4, with two subtle differences. Equation (3.15) allows vanishing margin $\rho = 0$ while the regularization term in Equation (3.13) prevents the margin (which is proportional to $\|\Lambda\|^{-2}$) from becoming zero. In contrast to the formulation in Equation (3.13), the margin in Equation (3.15) is not well defined in general. For separable data, for instance, ρ can be made arbitrarily large by rescaling the generalized distances. This is not considered critical for the experiments in the next section because Equation (3.15) is used in combination with I-smoothing.

3.3.4.2.3. Smooth Loss Functions and Limits

As mentioned above, we would like to use gradient-based optimization algorithms, which require differentiable training criteria. For this reason, we define smooth loss functions l_{γ} with the parameter to control the smoothness of the loss function. In addition to the differentiability, we require that $l_{\gamma}(\cdot) \xrightarrow{\gamma \rightarrow \infty} l(\cdot)$, i.e., the smooth loss function converges to the original loss function.

The hinge loss in Equation (3.14) is replaced with the soft-max approximation –

$$l_{\gamma}^{(hinge)}(c_n, d_n; \rho) = \frac{1}{\gamma} \log(1 + \sum_{c \neq c_n} e^{\gamma [-d_{nc} + \rho(\delta(c_n, c_n) - \delta(c, c_n))]}). \quad (3.16)$$

The Kronecker deltas $\delta(c_n, c_n) - \delta(c, c_n) = 1$ are added for illustration purposes below. As required, the approximation in Equation (3.16) converges to the hinge loss in Equation (3.14) for $\gamma \rightarrow \infty$.

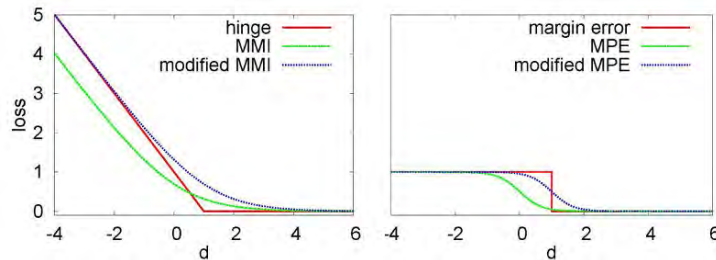


Figure 3.10: Comparison of loss functions for binary classification problem with $d = d_{nc}$. Left: comparison of hinge loss, MMI, and modified MMI ($\gamma = 1$). Right: comparison of margin error, MPE, and modified MPE ($\gamma = 3$).

We feel that the weak point about the hinge loss in pattern recognition is that it is not the measure used to evaluate the recognition systems eventually. This means that there is some guarantee regarding the generalization for the hinge loss, but not the recognition error. It is not clear how these two quantities are related. In addition, the hinge loss is not (optimally) robust against outliers because incorrectly classified observations are penalized proportionally to their distance from the decision boundary (see Figure 3.10). In contrast to the hinge loss function, error-based loss functions are bounded and thus, are expected to be more robust against outliers.

The ideal margin error is defined as -

$$l^{(error)}(c_n, d_n; \rho) := E(\hat{c}_\rho(d_n), c_n) \quad (3.17)$$

where $E(c, c')$ stands for some kind of error measure for c given c' . Here, the recognized class \hat{c}_ρ denotes the class that yields the maximum margin-posterior -

$$p_{\gamma\rho}(c|d_n) := \frac{e^{\gamma(-d_{nc} - \rho\delta(c, c_n))}}{\sum_{c'} e^{\gamma(-d_{nc'} - \rho\delta(c', c_n))}} \quad (3.18)$$

i.e., $\hat{c}_\rho(d_n) := \max_c \{p_{\gamma\rho}(c|d_n)\}$. In particular, the recognized class does not depend on the parameter (monotonic mapping of discriminant function). In the simplest case, this loss function counts the errors of the observations, $1 - \delta(\hat{c}_\rho(d_n), c_n)$.

We shall use this loss function similar to the MPE loss function -

$$l_\gamma^{(error)}(c_n, d_n; \rho) = \sum_c E(c, c_n) p_{\gamma\rho}(c|d_n). \quad (3.19)$$

Observe that for $\gamma \rightarrow \infty$, the margin-posteriors $p_{\gamma\rho}$ converge to the Kronecker delta $\delta(\hat{c}_\rho(d_n), c_n)$. This means that the loss function in Equation (3.19) converges to the true error $l^{(error)}$ in Equation (3.17) for $\gamma \rightarrow \infty$.

3.3.4.3 Modified Training Criteria in ASR

In this section, we apply the ideas from the last section to ASR. The SVM formulation in Section 3.3.4.2.1 better suits Hidden Conditional Random Fields (HCRFs) whereas the formulation in Section 3.3.4.2.2 is more intuitive for Gaussian HMMs (GHMMs). However, the equivalence of GHMMs and Gaussian-like HCRFs (Heigold 2008b) directly extends the first approach to GHMMs.

3.3.4.2.4. Choice of Margin

In ASR, the data is sequential. A natural extension of the margin from simple to sequential observations is to replace the above 1 - 0 margin indicated by the Kronecker delta in Equations (3.16) and (3.18) by the string accuracy $A(c, c_n)$ between c and c_n (Taskar 2003; Sha 2007). This means that the margin is proportional to the number of correct symbols. Unfortunately, there is no natural choice of accuracy. Two convenient definitions are the approximate accuracy also used for MPE and the frame-based accuracy (cf. Hamming distance). The advantage of these is that they allow for an efficient implementation, see (Heigold 2008a; Povey *et al.* 2008; Saon 2008) for further

details. Experiments suggest that the word error rate is rather insensitive to the choice of margin (Heigold 2008a).

In the remainder of this section, different variants of the smooth optimization problem introduced in the last section are described and experimental results are presented. The resulting update rules and gradients have the same structure as for conventional MMI and MPE, and differ from conventional MMI and MPE only in the scores that enter the forward/backward algorithm (Heigold 2009). For this reason, the corresponding update rules and gradients are not explicitly stated.

3.3.4.2.5. Tasks and Experimental Setups

The different tasks are summarized in Table 3.20.

RWTH Basic Setup: The basic setups have been described by Heigold (2009). The GHMMs use CART-tied triphones and globally pooled variances. This allows us to produce rather good Maximum Likelihood (ML) baselines consisting of a fairly high number of densities, cf. Table 3.20. The LVCSR tasks use all Vocal Tract Length Normalization (VTLN) and Speaker Adaptive Training (SAT). In addition, some setups use Multi-Layer Perceptron (MLP) features. Except for EPPS (EBW), Rprop (Riedmiller 1993) was used for the optimization of the discriminative models. The best training iteration was determined on the development corpus.

IBM Basic Setup: The basic setup is the same as that used by Povey *et al.* (2008) and Saon (2008). All systems have state-specific variances; all systems have quinphone cross-word context; all have cepstral mean subtraction and those with VTLN have cepstral variance normalization. All setups with fMLLR are SAT trained. Setups that are shown as having fMLLR are also tested with regression tree MLLR following fMLLR. The optimization of the discriminative models was done with different versions of the EBW algorithm (4 iterations).

Identifier	Corpus description	Train/Test [h]	Features/Model	#States/#Dns
SieTill	German digit strings	11.3/11.4	25 LDA(MFCC)	215/27k
EPPSE	English parliament plenary sessions	90/2.9 (Eval07)	45 LDA(MFCC+voicing) +VTLN+SAT	4,500/830k
CBNBC230	Chinese BN & BC	230/2.9 (Eval06)	45 LDA(MFCC+tone) +VTLN+SAT	4,500/1,100k
CBNBC1500	Chinese BN & BC	1,500/2.9 (Eval06)	45 LDA(PLP+voicing)+3 tones +32 MLP+VTLN+SAT	4,500/1,200k
ABN1400	Arabic BN	1,400/Dev07	VTLN+fMLLR	5,000/400k
ACTS80	Iraqi conversational telephone speech	80/2.5	VTLN+fMLLR	3,500/75k
EBN50	English BN	50/RT-04	Unadapted	2,200/50k
EBN700	English BN	700/RT-04	Unadapted	6,000/250k

Table 3.20: Corpora and setups, BN (broadcast news), BC (broadcast conversation).

3.3.4.2.6. Large Margin MMI

Here, different variants of the large margin optimization problem based on the hinge loss are presented. We adjust the notation slightly: W denotes a word sequence and X the

sequence of feature vectors, $p(W)$ represents the language model, and $p(X|W)$ stands for the acoustic model (without scaling factors for simplicity).

This criterion is based on the large margin formulation in Section 3.3.4. Modified MMI is defined as

$$\mathcal{F}_\gamma^{(MMI)}(\Lambda) = -\log p(\Lambda) - \frac{J}{N} \sum_{n=1}^N \frac{1}{\gamma} \log \left(\frac{[p(W_n)p_\Lambda(X_n|W_n)e^{-\rho A(W_n,W_n)}]^\gamma}{\sum_W [p(W)p_\Lambda(X_n|W)e^{-\rho A(W,W_n)}]^\gamma} \right)$$

For Gaussian-like CRFs, the acoustic model is log-linear and $\log p(\Lambda)$ stands for the L2-norm regularization as in Equation (3.13). For this choice, it can be shown that for $\gamma \rightarrow \infty$, the optimization problem converges to the SVM optimization problem in Section 3.3.4 using the hinge loss function.

In a control experiment, modified MMI was applied to the SieTill task (see Table 3.20). This simple digit string recognition task severely suffers from overfitting. As expected, the margin term helps significantly. For the best log-linear HMM, the ML baseline digit error rate was 1.81% whereas conventional and modified MMI yielded 1.77% and 1.59%, respectively. Similar effects were observed on other log-linear setups, see (Heigold 2008a).

Results for conventional GHMMs and the phoneme error as loss function can be found in Section 3.3.4.2.7.

Boosted MMI (BMMI): Boosted MMI (BMMI) is based on the same theoretical framework as modified MMI in the previous section. BMMI uses conventional GHMMs for the acoustic model and I-smoothing as regularization. The accuracy A for the margin is the approximate phoneme accuracy known from standard MPE.

In practice and in addition to the boosting factor, BMMI differs from standard MMI by the way the learning rate constants (Povey 2008) are set (canceling of any shared part of the numerator and denominator statistics on each frame, denoted by suffix ‘-c’), and the acoustic model chosen for I-smoothing (previous iteration, denoted by suffix \rightarrow prev). Here, we shall focus on the effect of the boosting term. The interested reader is referred to (Povey 2008) for more results. The results are summarized in Table 3.21 columns 2 and 3. These experiments suggest that some improvement is due to the boosting term while the greater improvement is probably due to the refined learning rate constants and I-smoothing (Povey 2008). The results appear to be rather sensitive to the choice of optimization.

BMMI can also be used for feature-space discriminative training, e.g., fMPE or fBMMI (see Table 3.21, column 5). Here, it is not clear what the effect of the boosting term is.

Criterion	WER [%]		Criterion	WER [%]	Criterion	WER [%]	
	EBN50	EBN700				EBN50	ABN1400
ML	25.3	20.5	ML	43.2	ML	25.3	17.1
MMI		18.9	fBMMI-c	37.5	MMI-c→prev	22.4	15.2
MMI-c	22.4		+BMMI-c	35.9	BMMI-c→prev	21.3	
BMMI		18.6	fMPE	40.4	M-BMMI-c →prev	21.2	14.2
BMMI-c	21.7	17.9					
MPE	21.6	18.1	+MPE	36.8	MPE	21.6	

Table 3.21: Word Error Rate (WER) for EBN50 and EBN700 (RT04), ACTS80, and ABN1400 (see text for meaning of suffixes ‘-c’ and ‘→prev’).

Margin BMMI (M-BMMI): The approach of Saon (2008) is based on the large margin formulation in Section 3.3.4.2.2 in combination with the smooth hinge loss -

$$\mathcal{F}_\gamma^{(M-MM)}(\Lambda, \rho) = -\rho - \frac{1}{N} \sum_{n=1}^N \frac{1}{\gamma} \log \left(\frac{[p(W_n)p_\Lambda(X_n|W_n)e^{-\rho A(W_n, W_n)}]^\gamma}{\sum_W [p(W)p_\Lambda(X_n|W)e^{-\rho A(W, W_n)}]^\gamma} \right)$$

M-BMMI uses the state-based Hamming accuracy A for the margin term. This is in contrast to BMMI (approximate phoneme accuracy, see above). The Gaussian parameters Λ are optimized with EBW as described by Povey (2008) and Saon (2008) while ρ is kept fixed. Table 3.21, column 7 and 8 compares M-BMMI with some other training criteria. The proposed approach to margin-based training is not restricted to the optimization of GHMMs. Semi-tied covariance transforms, for instance can be estimated as well in this framework (Saon 2009).

Comparison of MMI Variants: The three variants of large margin MMI are very similar. Modified MMI and BMMI are identical on the theoretical side, but differ in the optimization procedure (Rprop vs. highly tuned EBW). M-BMMI and BMMI use the same optimization strategy. M-BMMI and BMMI/Modified MMI differ in the choice of the accuracy for the margin (state based Hamming accuracy vs. approximate phoneme accuracy) and the regularization. All three variants tune the margin parameter manually (although not necessary from a theoretical point of view). Under these assumptions, M-BMMI and BMMI are identical except for the choice of accuracy for the margin.

3.3.4.2.7. Modified MPE

In contrast to the hinge loss, the recognition error is bounded as illustrated in Figure 3.10. The approximate phoneme error is expected to be closer to the true word error and to be more robust against outliers (e.g., imperfect transcriptions).

We define an MPE-like criterion to be minimized representing a smoothed margin phoneme error with regularization according to Section 3.3.4

$$\mathcal{F}_\gamma^{(MPE)}(\Lambda) = -\log p(\Lambda)$$

$$+ \frac{J}{N} \sum_{n,W} E(W, W_n) \left(\frac{[p(W_n)p_\Lambda(X_n|W)e^{-\rho A(W, W_n)}]^\gamma}{\sum_{W'} [p(W')p_\Lambda(X_n|W')e^{-\rho A(W', W_n)}]^\gamma} \right)$$

Using log-linear acoustic models and L2-norm regularization, it can be shown that for $\gamma \rightarrow \infty$ this criterion converges to the SVM optimization problem in Section 3.3.4 using the associated margin phoneme error (Heigold 2008a). As for boosted MMI, we can use GHMMs for $p_\Lambda(X|W)$ and I-smoothing for regularization. The accuracy A for the margin is the same approximate phoneme accuracy as for MPE.

The results for modified MPE are summarized in Table 3.22.

Criterion	WER [%]		
	EPPSE	CBNBC	
		230	1500
ML	12.0	21.9	17.9
MPE	11.5	20.6	16.5
Modified MPE	11.3	20.3	16.3

Table 3.22: Word Error Rate (WER) for EPPSE (Eval07) and CBNBC230/CBNBC1500 (Eval06).

Justification of MPE Heuristics: Observe that formally, modified MPE is similar to conventional MPE. Indeed, modified MPE gives some new insight into several heuristics typically used for conventional discriminative training: the smoothing parameter corresponds with the scaling factor for the posteriors, I-smoothing is a refined regularization term (Povey 2003), and the weak language model can be considered an approximation of the margin term.

3.3.4.4 Discussion

We discussed approaches to modifying existing training criteria, e.g., MMI and MPE to include a margin term. These modified criteria are closely related to standard large margin classifiers, e.g., SVMs. This approach to large margin optimization has the advantage that it fits into the existing framework used for conventional discriminative training and allows for a direct evaluation of the utility of the margin term. As expected, modified MMI outperforms conventional MMI on a simple digit recognition task where over-fitting is an issue ($\geq 75\%$ of total discriminative improvement). For LVCSR, however, the additional margin term is less important, compared with the best state-of-the-art systems ($\leq 25\%$ of total discriminative improvement). Reasons for this outcome might be that the loss term dominates for LVCSR, and that the margin concept is already well approximated by several heuristics (e.g., weak language model) used for conventional discriminative training.

3.3.5. Approaches to Speech Recognition based on Speaker Recognition Techniques

Authors: Daniel Povey, Stephen M. Chu, Jason Pelecanos and Hagen Soltau

3.3.5.1 Introduction

A basic approach long used in text-independent speaker recognition is to train a Gaussian Mixture Model (GMM) for each speaker and to verify a test utterance based on the likelihood of the utterance given that speaker's GMM compared with a cohort of other models or a single "universal background model". A drawback of this approach is that it is hard to robustly estimate the parameters of the speaker-specific GMMs given limited data. Therefore, in recent years, approaches have been devised to estimate the parameters more robustly based on Maximum A Posteriori (MAP) (Gauvain *et al.* 1994; Reynolds *et al.* 2000) and subspace methods such as nuisance attribute projection (Solomonoff *et al.* 2005), within-class covariance normalization (Hatch *et al.* 2006) and joint factor analysis (Kenny *et al.* 2008). This joint factor analysis method in particular not only accounts for cross-speaker variability, but also directly addresses the issue of within speaker variability such as telephone type and background noise. The same problems that exist in speaker recognition, such as estimation given limited data and irrelevant sources of variation, also exist in speech recognition so it is natural to try to use those speaker recognition techniques in a speech recognition context. We have experimented with two different approaches.

3.3.5.2 MAP-Based Adaptation

In *Boosted MMI for Feature and Model Space Discriminative Training* (Povey *et al.* 2008), we introduced a speech recognition technique based on MAP adaptation. Note that although we use the term "adaptation" it is not a speaker adaptation technique; the term refers to the adaptation of models to particular phonetic states. We first train a large mixture of Gaussians (e.g., 750 Gaussians), which we refer to as a "Universal Background Model" (the term is borrowed from speaker recognition). We then MAP adapt this large GMM to each clustered phonetic state. In order to improve the smoothing we actually used a modified form of MAP estimation, which took into account the phonetic decision tree, so that a phonetic state is adapted first to phonetic states that are close in the tree. The adapted models were diagonal covariance Gaussians, but we also introduced a separate Semi-tied Covariance (STC) transformation (Gales 1998) for each of the 750 Gaussians. This helped our experimental system, and in experiments performed later on we were unable to obtain any improvement from this type of technique in our baseline setup. In order to make the memory usage manageable, we introduced a parameter pruning scheme tied to the tree structure.

Table 3.23 shows the results; Povey *et al.* (2008a) present additional details. The table shows about 1.5% absolute improvement for all conditions, but the situation is a little more complicated than that. The baselines with fMLLR and MLLR are "normal size" baselines with 500,000 Gaussians (about the size we would build for a normal evaluation system), but the VTLN-only baseline is twice the normal size at one million

Gaussians, which helped by 0.8%. This masks the fact that the MAP based system gave much more improvement in the VTLN-only condition. We increased the size of this baseline to provide a fairer comparison to the MAP based system which has a very large number of (smoothed) parameters, several times more than even the largest baseline system. Due to time constraints, we did not rerun the fMLLR and fMLLR+MLLR baselines with the larger size, which would have reduced our improvement. Regardless of the exact amount of improvement from the MAP based system, we did not pursue this approach because we believed that it would be too hard to combine this style of system with discriminative training. This is due to its very large number of parameters, which does not combine well with discriminative training (Povey 2004).

Adaptation	Baseline	MAP	MAP+STC
VTLN	17.9	17.5	16.6
+fMLLR	16.6		15.0
+MLLR	16.2		14.8

Table 3.23: Results from MAP-based adaptation: Mandarin, dev'07.

3.3.5.3 Subspace Adaptation

The style of system we are currently working with is related to the joint factor analysis approach of Kenny *et al.* (2008). In this approach, the means of a large mixture of Gaussians are concatenated to form a “supervector”, and we allow the parameters of our adapted system to vary in a subspace of that vector space, so each phone in context would be represented by a vector in our chosen subspace. For example, we have used a subspace dimension of 50 in a system with feature dimension 40 and 750 Gaussians in the GMM, so we are using 50 out of a possible 30,000 dimensions. We also use a separate subspace to represent the nuisance effect of speaker variation, which is analogous to the use of a separate subspace to represent the nuisance effect of channel or session variation as shown by Solomonoff *et al.* (2005) and Kenny *et al.* (2008). We have added certain extensions to this basic model. One is to replace the use of a single vector to represent each phone in context, with a set of vectors each with their own weights. Each vector represents an adapted mixture of Gaussians, so our extended model represents a mixture of mixtures of Gaussians. We also make the Gaussian mixture weights vary as a function of our subspace vector. The log mixture weights vary linearly with the subspace vector, and we normalize them to sum to one. A further feature of our model is that each of the e.g., 750 basic Gaussians has a full covariance. The covariances are shared between the classes, and are not adapted as in the MAP based approach described above.

3.3.5.3.1. The Model

Here we describe our approach only briefly; see “Subspace Gaussian Mixture Models for Speech Recognition” (Povey 2009) for further details. For each acoustic state j and given the speaker s , the probability model $p(\mathbf{x}|j)$ is:

$$p(\mathbf{x}|j) = \sum_{i=1}^I w_{ji} \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_{ji}^{(s)}, \boldsymbol{\Sigma}_i) \quad (3.20)$$

$$\boldsymbol{\mu}_{ji}^{(s)} = \mathbf{M}_i \mathbf{v}_j^+ + \mathbf{N}_i \mathbf{v}^{+(s)} \quad (3.21)$$

$$w_{ji} = \frac{\exp(\mathbf{w}_i^T \mathbf{v}_j^+)}{\sum_{i'=1}^I \exp(\mathbf{w}_{i'}^T \mathbf{v}_j^+)}, \quad (3.22)$$

so we describe the acoustic state by the vector \mathbf{v}_j and the speaker by $\mathbf{v}^{(s)}$; the notation \cdot^+ means appending a 1 to the vector (to handle constant offsets). These vectors have a dimension of about 50. I is the number of Gaussians in the model being adapted and will typically be about 750. The acoustic states $1 \leq j \leq J$ have the normal range, e.g., several thousand, although this style of system can give its best results with more states than the baseline. The parameters \mathbf{M}_i and \mathbf{N}_i describe the phonetic subspace and the speaker subspace. The weights w_{ji} are a log-linear function of the model weights, normalized to sum to one. The parameters \mathbf{w}_i control the projection from the model subspace to the weights. The covariance matrices $\boldsymbol{\Sigma}_i$ are full covariance matrices that are not specific to the acoustic state but to the Gaussian index i . Experiments with a subspace representation of the inverse (diagonal) covariance matrix, combined with semi-tied covariance transform to handle more general rotations, failed to show any improvements. On top of the basic model described above, we use the notion of a sub-state so that a state can be represented by a weighted sum over the models given by different subspace vectors.

The speaker adaptation in this model involves training very few speaker-specific parameters (we use a subspace dimension of typically about 50), so our speaker adaptation is actually done on a per utterance basis. We combine this with conventional fMLLR/constrained MLLR, which is straight-forward, and also with MLLR, which is less straight-forward because it does not combine well with the efficient model evaluation (we have to recompute the normalizing factors for each speaker). For the Arabic results that we report here, we also include an extension of the speaker adaptation technique that uses more speaker-specific parameters, but we do not describe this here in detail.

3.3.5.3.2. Likelihood Evaluation

When we evaluate likelihoods given the model, we can arrange the computation so that computing the likelihood for a particular acoustic state j and Gaussian index i involves no more than a dot product in the model subspace. We can then prune the Gaussian indices i to compute only a small subset of the, say, 750 Gaussians. This computation requires us to store a normalizing term for each state j and Gaussian index i , and this normalizing term dominates the memory requirements of the model. Explicitly constructing the projected means in this model is impractical due to memory constraints. The model is about two to four times slower to decode with than a normal system, but this gap could easily be eliminated without losing much performance.

3.3.5.3.3. Training with the Subspace Model

The process of model training consists of training the global parameters \mathbf{M}_i , \mathbf{N}_i and \mathbf{w}_i and the state-specific parameters \mathbf{v}_j . We initialize the vectors \mathbf{v}_j randomly and then

iteratively optimize the projections \mathbf{M}_i and \mathbf{w}_i and the vectors \mathbf{v}_j . The speaker subspace is trained in a similar way; after training the main model we initialize the projections N_i and then iteratively re-estimate the speaker vectors $\mathbf{v}^{(s)}$ and the projections \mathbf{N}_i . In general, the statistics required to re-estimate these parameters are quite compact and the re-estimations are fairly straightforward; we get an auxiliary function, which is quadratic in the parameters we are trying to optimize, and it is easy to optimize. Training requires about the same number of iterations as a normal system (about 30), but the first 10 iterations are done using only one pass over the data because prior to introducing the sub-states, it is possible to store appropriately pruned statistics, which allow us to do many iterations of update, without re-accessing the data.

The only part of the optimization, which is not absolutely straightforward, is the part which relates to the weight projections \mathbf{w}_i and the effect of the weights on the model vectors \mathbf{v}_j . This is handled with a suitable quadratic approximation and is not a problem in practice.

3.3.5.3.4. Discriminative Training

We were able to combine discriminative training with this style of system. Model-space discriminative training is fairly straightforward, as we can use a version of the Extended Baum-Welch type of technique using weak sense auxiliary functions (Povey 2004). We did introduce some details that are specific to this framework. We use a factor in the learning rate for the state-specific vectors that slows down learning when the discriminative data counts are very small. More importantly, we find that we tend to get an instability that is localized to particular Gaussian indices i so we introduced a method to detect this via differences in the (appropriately normalized) gradient directions on successive iterations; if we detect instability we slow down the learning rates for the affected matrices. The discriminative training we implemented is based on Minimum Phone Error (MPE) (Povey and Woodland 2002) and Boosted MMI (Povey *et al.* 2008) objective functions. The tuning is quite similar to a normal system, e.g., $E = 2$, four to six iterations required; with E defined by analogy with (Povey 2004). Lattices were generated from the ML version of our factor-analyzed system, although this presented some challenges due to the slower decoding speed.

Feature space discriminative training, e.g., fMPE (Povey 2005) is fairly straightforward as it only requires the calculation of the model likelihood gradients w.r.t. the features. We implemented a simpler version than the one described in *Improvements to fMPE for Discriminative Training of Features* (Povey 2005) as we omitted the “indirect differential” and did not perform the ML model update in between each feature-space update. These two things are related as the “indirect differential” is intended to account for the effect of the model update. Our typical setup is to do ML training, then model space discriminative training, then feature space discriminative training, although for the Arabic system we used a more complicated sequence. Our experience is that model space discriminative training works about as well as for a normal type of system, but feature space discriminative training gives much less additional improvement. One plausible reason for this is that the structure of the transformation used in fMPE is quite similar to the model we are using (it is based on a similar sized mixture of Gaussians), so

there may be less synergy between the feature and model space than for a normal system. Despite this, the improvement at the ML level tends to be large enough to outweigh the reduced benefit from discriminative training.

3.3.5.3.5. Probability Scale

An interesting aspect of this kind of model (and it also applies to the MAP-adapted type of model), is that the optimal acoustic weight is very different from our normal system, around 1/12 as opposed to our normal value of 1/19. Because our features consist of nine frames spliced together and projected with LDA, and because LDA can be interpreted as an ML technique, we can argue that the “true” acoustic weight is (or should be bounded in some direction by) 1/9. This is because if we were to skip nine frames each time, we would be modeling each un-spliced frame exactly once. It is encouraging that this style of model takes us closer to the theoretically motivated acoustic weight.

3.3.5.3.6. Details and Tuning of Training

We start training by initializing the un-adapted GMM, which typically has 750 full covariance Gaussians and is trained on all speech classes mixed together. We then accumulate count and first order statistics using the product of these 750 Gaussians' posteriors and the (zero-one) phonetic class posteriors, based on a Viterbi alignment of the data using a baseline system. These statistics are compressed to fit in memory by discarding small counts. We perform the first 10 or so iterations of training at once without re-accessing the data, using these statistics. This is sufficient to get a good initial estimate of the model. From this point we can start increasing the number of sub-states and doing further passes over the data while computing Gaussian posteriors based on the adapted versions of the models. In these iterations of training, we accumulate statistics that are the same size as the parameter set (representing the linear term in the objective function for each parameter), plus Gaussian-specific counts, which determine the quadratic parts of the objective function. These Gaussian-specific counts dominate the statistics in terms of memory.

If we are performing speaker adaptation as part of this approach, we compute on each training iteration the speaker-specific (actually, utterance-specific) vectors as we access each training file.

3.3.5.3.7. Results

English broadcast news, 50h training:

We first show results on 50 hours of training data, on an English Broadcast News task. This 50 hours corpus is a subset of the Hub4 data which we have previously published results on (Povey *et al.* 2008). Testing is on the RT'04 English Broadcast News test set.

The results in Table 3.24 are quite spectacular without discriminative training (top row); nearly 20% relative improvement from this approach. However, the improvement after discriminative training (bottom row) is only 5% relative. This situation where we have relatively little data plays well to the strengths of this system which has relatively

few parameters (about half the baseline). We do not have exactly comparable numbers on this testing setup for the MAP-based approach described above, but results on the same training and test sets using a slightly different configuration showed an improvement of 24.6% to 23.6% with only ML training. It seems that with this amount of training data the subspace approach does much better than the MAP-based approach.⁸ However, the difference between the MAP and factor analyzed approaches was less clear with the larger Mandarin setup, which we used for MAP experiments Table 3.23. Again, we do not have exactly comparable numbers, but we estimate that the two systems in their best configuration would give about the same results under ML training.

Conditions:	Baseline	Subspace
VTLN+fMLLR+MLLR	24.3%	19.6%
+fMMI+MMI	18.2%	17.3%

Table 3.24: Subspace-adapt vs normal system on 50 hours English BN (test: RT'04).

Arabic January 2009 Evaluation:

Table 3.25 shows the word error rate numbers for our Arabic system prepared for the January 2009 GALE evaluation. The VxU refers to a vowelized (V) Arabic system cross-adapted on the output of an unvowelized (U) system. “vfr” refers to “variable frame rate,” an experimental technique, which we applied in test time to normalize speaking rate. The sub-space (SUB) system is built on top of a vowelized system, so we cross adapt to the unvowelized output giving us SUBxU. The results shown are the final numbers, with all forms of adaptation plus discriminative training. The discriminative training regime applied to the subspace system was rather complicated and involved six iterations of model-space boosted MMI training, lattice regeneration, four further passes of boosted MMI, five iterations of fMPE and four iterations of MPE. Normally we would expect a system to stop improving after so much discriminative training, but this system proved quite robust to it. The discriminative training regime was applied *ad hoc*; we do not believe that there is anything special about this particular setup. The improvements versus our baseline system were about 1% absolute before discriminative training, i.e., larger than our final improvements with discriminative training included.

System:	Dev07	Dev08	Eval07 (unsequestered)
VxU.vfr	10.0%	11.5%	14.4%
SUBxU.vfr	9.5%	11.1%	14.2%

Table 3.25: Arabic evaluation system, January 2009.

⁸ Note that these numbers were obtained without MAP-adapting the variances, since the variance adaptation hurts on this smaller training set.

Mandarin January 2009 Evaluation:

We also trained this style of system for IBM's Mandarin submission to the January 2009 evaluation, trained on 1737 hours of data; see Table 3.26 for the results. Here, “T” means a tonal system (i.e., with tone features) and “N” means a non-tonal system, so the TxN baseline is a tonal system adapted (with fMLLR and MLLR) on the text output of a non-tonal system. The subspace (SUB) system is built on tonal features, and adapted on the non-tonal system. The baseline discriminative training regime was with five iterations of feature-space boosted MMI (fBMMI) followed by tree rebuilding and four iterations of model-space boosted MMI. The subspace system had five iterations of model-space BMMI and five of feature-space BMMI. The results with the subspace are not clearly better than the baseline; however, we need to consider the improvement we get from tree-based MLLR for cross-adaptation, which we implemented for the baseline, but not the subspace system. Using our default, non-optimized min-count of 3000 (vs. 1250 in Table 3.26), our baseline numbers on Dev07 and Eval07(unsequestered) were 9.9% and 9.6% respectively, which shows some of the effect of tree-based MLLR. So, if we had implemented tree-based MLLR for the subspace system, we believe we would have been able to show a clearer improvement over the baseline.

System:	Dev07	Dev08	Eval07 (unsequestered)
TxN.vfr	9.7%	8.6%	9.2%
SUBxN.vfr	9.7%	8.3%	9.4%

Table 3.26: Mandarin evaluation system, January 2009.

3.3.5.4 Conclusion

We have briefly described our work with MAP-based adaptation and subspace adaptation (joint factor analysis) for speech recognition. In these techniques we apply robust model adaptation techniques originally devised for adapting to speakers, but instead apply them to adapting to phonetic states. In both cases we get improvements under ML training compared with a conventional system, when trained with both small and large amounts of data. The first, MAP-based technique has the drawback that it has a very large number of parameters and, therefore, presents problems for discriminative training. The second, joint factor analysis based technique is more promising. We have demonstrated word error rate improvements on a large task in an evaluation setting; however, the improvements are quite modest. In addition to the word error rate improvements, the framework is quite useful in terms of allowing new kinds of techniques to be introduced. It may also offer the opportunity to improve speaker recognition techniques as it represents a kind of “unified model” of speaker and phonetic-state variation. The main challenge is the complexity of the approach, which may limit its uptake. Further work based on these ideas can be found, for example, in “Subspace Gaussian Mixture Models for Speech Recognition” (Povey *et al.* 2010).

3.4 Language Models

3.4.1. Introduction

Author: Katrin Kirchhoff

Over the last decade, language modeling has moved from standard generative Markov model style n -gram modeling to models incorporating more structure and different probability estimation techniques, such as syntax-based language models, neural language models, or discriminative language models. These models have been used both for speech recognition and for machine translation. The contributions in this chapter address the use of such models in GALE speech-to-text and machine translation systems, as well as, the problem of language model adaptation.

One section in this section is based on the neural network language modeling approach. Neural network language models (Bengio 2001) (also called continuous space language models) map the discrete word representation commonly used in text processing to a continuous representation and estimate n -gram probabilities in this continuous space. The motivation for this approach is that a continuous representation is likely to lead to smoother probability estimates, which in turn allows better generalization to unseen contexts. The mapping is performed by a neural network with a dedicated architecture that includes a mapping (or projection) layer in addition to the standard hidden and output layers. The weights of this layer are shared among all words in the vocabulary and are trained jointly with all other network parameters during language model training.

Section 3.4.2 describes the use of neural language models in their GALE Mandarin and Arabic speech recognition systems. The section reports on various ways of increasing the efficiency of this approach.

Section 3.4.3, by Tam *et al.* (CMU), addresses the problem of topic adaptation. Starting from the framework of latent Dirichlet allocation (LDA) (Blei 2003) to model topic structure, the section proposes several extensions: the basic LDA approach is generalized to a tree-based model that takes into account hierarchical correlations between different topics, and it is enriched with bigram information. Finally, a bilingual topic adaptation technique based on latent semantic analysis is developed to be able to adapt different types of models (language models, and also, translation models) cross-lingually. The proposed techniques were evaluated within GALE Chinese and Arabic speech recognition and translation systems and showed significant improvements.

3.4.2. On the Use of Continuous Space Neural Network Language Models

Authors: Holger Schwenk, Abdelkhalek Messaoudi, Jun Luo and Jean-Luc Gauvain

3.4.2.1 Introduction

Over the last years there has been growing interest in using neural networks for language modeling. Instead of relying on a backoff component, the neural network

approach attempts to overcome the data sparseness problem by performing the estimation in a continuous space. In early work, this type of language model was mostly used for tasks for which only a very limited amount of in-domain training data is available. In the context of the EARS and GALE programs new algorithms were developed to train a neural network language model on very large text corpora. This made possible the use of the approach in domains where several hundreds of millions words of texts are available. The neural network language model has been evaluated in state-of-the-art continuous speech recognizers for several languages and tasks. Word error reductions on the order of 0.5% to 1% absolute have been obtained.

The basic idea of the neural network LM, also called continuous space LM, is to project the word indices onto a continuous space and to use a probability estimator operating on this space (Bengio and Ducharme 2001). Since the resulting probability functions are smooth functions of the word representation, better generalization to unknown n -grams can be expected. A neural network can be used to simultaneously learn the projection of the words onto the continuous space and to estimate the n -gram probabilities. This is still an n -gram approach, but the LM posterior probabilities are interpolated for any possible context of length $n - 1$ instead of backing off to shorter contexts. The architecture of this approach is described in detail in Section 3.4.2.2.

To date, this new language modeling method has been applied to large vocabulary speech recognition in six different languages (English, French, Spanish, Arabic, Mandarin and Dutch) achieving systematically consistent improvements in the word error rate. We describe here the work performed in the framework of the GALE project to use this technology in our Arabic and Mandarin speech recognition systems. In addition, the continuous space language model was used in a statistical machine translation system based on the open-source Moses decoder. Improvements of about one point BLEU were observed in a large Arabic/English translation systems (Schwenk and Esteve 2008).

3.4.2.2 Architecture of the Continuous Space LM

The basic idea of the continuous space LM (CSLM) is to project the word indices onto a continuous space and to use a probability estimator operating on this space (Bengio and Ducharme 2001). Since the resulting probability functions are smooth functions of the word representation, better generalization to unknown n -grams can be expected. A neural network can be used to simultaneously learn the projection of the words onto the continuous space and to estimate the n -gram probabilities. This is still a n -gram approach, but the LM posterior probabilities are “interpolated” for any possible context of length $n - 1$ instead of backing off to shorter contexts.

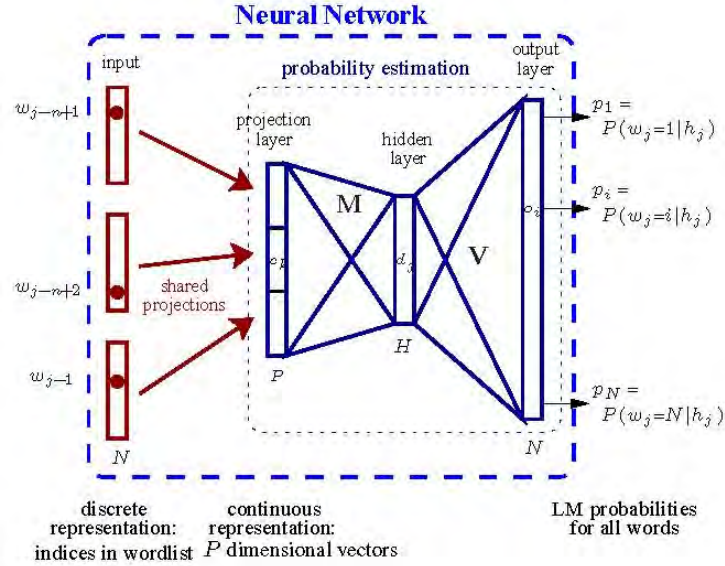


Figure 3.11: Architecture of the continuous space language model. h_j denotes the context $w_{j-n+1}, \dots, w_{j-1}$. P is the size of one projection and H and N is the size of the hidden and output layer respectively. When shortlists are used the size of the output layer is much smaller than the size of the vocabulary.

The architecture of the continuous space n -gram LM is shown in Figure 3.11. A standard fully-connected multi-layer perceptron is used. The inputs to the neural network are the indices of the $n-1$ previous words in the vocabulary $h_j = w_{j-n+1}, \dots, w_{j-2}, w_{j-1}$ and the outputs are the posterior probabilities of *all* words of the vocabulary:

$$P(w_j = i | h_j) \quad \forall i \in [1, N] \quad (3.23)$$

where N is the size of the vocabulary. The input uses the so-called 1-of- n coding, i.e., the i^{th} word of the vocabulary is coded by setting the i^{th} element of the vector to 1 and all the other elements to 0. The i^{th} line of the $N \times P$ dimensional projection matrix corresponds to the continuous representation of the i^{th} word. Let us denote c_k these projections, d_j the hidden layer activities, o_i the outputs, p_i their softmax normalization, and m_{jl} , b_j , v_{ij} and k_i the hidden and output layer weights and the corresponding biases. Using these notations the neural network performs the following operations:

$$d_j = \tanh(\sum_l m_{jl}c_l + b_j) \quad (3.24)$$

$$o_i = \sum_j v_{ij}d_j + k_i \quad (3.25)$$

$$p_i = e^{o_i} / \sum_{k=1}^N e^{o_k} \quad (3.26)$$

The value of the output neuron p_i corresponds directly to the probability $P(w_j = i|h_j)$. Training is performed with the standard backpropagation algorithm minimizing the following error function:

$$E = \sum_{i=1}^N t_i \log p_i + \beta (\sum_{jl} m_{jl}^2 + \sum_{ij} m_{ij}^2) \quad (3.27)$$

where t_i denotes the desired output, i.e., the probability should be 1.0 for the next word in the training sentence and 0.0 for all the other ones. The first part of this equation is the cross-entropy between the output and the target probability distributions, and the second part is a regularization term that aims to prevent the neural network from over-fitting the training data (weight decay). The parameter β has to be determined experimentally.

It can be shown that the outputs of a neural network trained in this manner converge to the posterior probabilities. Therefore, the neural network directly minimizes the perplexity on the training data. Note also that the gradient is back-propagated through the projection-layer, which means that the neural network learns the projection of the words onto the continuous space that is best for the probability estimation task.

This basic architecture has a quite high complexity. The following methods were deployed resulting in processing times of less than 0.1xRT:

1. *Shortlists*: the neural network is only used to predict the LM probabilities of a subset of the whole vocabulary.
2. *Lattice rescoring*: speech recognition is done with a standard back off LM and a word lattice is generated. The CSLM is then used to rescore the lattice.
3. *Grouping*: all LM probabilities needed for one lattice are collected and sorted. By these means all LM probability requests with the same context h_t lead to only one forward pass through the neural network.
4. *Block mode*: several examples are propagated at once through the neural network, allowing the use of faster matrix/matrix operations.
5. *CPU optimization*: machine specific BLAS libraries are used for fast matrix and vector operations.
6. *Resampling*: instead of cycling sequentially through all the data, we resample examples from the large corpora.

The idea behind shortlists is to use the neural network only to predict the s most frequent words, $s \ll |V|$, reducing by these means drastically the complexity. All words of the word list are still considered at the input of the neural network. The LM probabilities of words in the shortlist (\hat{P}_N) are calculated by the neural network and the

LM probabilities of the remaining words (\hat{P}_B) are obtained from a standard 4-gram backoff LM:

$$\hat{P}(w_t|h_t) = \begin{cases} \hat{P}_N(w_t|h_t)P_S(h_t) & \text{if } w_t \in \text{shortlist} \\ \hat{P}_B(w_t|h_t) & \text{else} \end{cases} \quad (3.28)$$

$$P_S(h_t) = \sum_{w \in \text{shortlist}(h_t)} \hat{P}_B(w|h_t) \quad (3.29)$$

It can be considered that the neural network redistributes the probability mass of all the words in the shortlist. This probability mass is precalculated and stored in the data structures of the backoff LM. A backoff technique is used if the probability mass for a requested input context is not directly available. A detailed description of the speed-up techniques is given by Schwenk (2004) and Schwenk and Gauvain (2005b).

Although some fast training algorithms were already proposed (Schwenk 2004), using a standard fully connected multi-layer perceptron, training on very large corpora can be time consuming. Therefore, an algorithm was proposed by Schwenk and Gauvain (2005b) to enable training the neural network on any arbitrarily large corpus. The basic idea is quite simple and consists of selecting small random subsets of the data for each epoch rather than using all the available data. This procedure has the advantages that there is no limit on the amount of training data; that after several epochs, most of the examples have been seen at least once; and that changing the examples after each epoch may potentially increase the generalization performance. This procedure has been used with more than three billion words of training data.

3.4.2.3 Experimental Evaluation in ASR

Word error reductions using a continuous space LM for LVCSR were first reported on the EARS conversational telephone speech recognition task (Schwenk and Gauvain 2002). Since then significant improvements have been reported on many speech recognition tasks and languages. Here results are reported on GALE broadcast data for both Arabic and Mandarin.

Continuous space language models were trained using a 290k word list for Arabic and a 56k word list for Mandarin. The short list was of size 12k for Arabic and 8k for Mandarin. A traditional 4-gram language model was trained on all the available data for each language and pruned with high pruning threshold, and used as a short list backoff model to calculate the probability masses and to cover the probabilities of words not included in the short list.

Since training the neural network is quite long, the text data was sampled according to the size of the corpus and its importance. For Arabic, the manual transcriptions of the audio data and the web transcripts are used in their entirety. Across the text sources a total of about 26 million contexts are projected, and each training iteration takes about a day. Table 3.27 shows the approximate number of contexts by Arabic text source for a training iteration.

<i>Text source</i>	<i>#contexts</i>	<i>Text source</i>	<i>#contexts</i>	<i>Text source</i>	<i>#contexts</i>
trans+webstrans	16.6 M	addustour	0.4 M	akhbar	0.3 M
Nhr	2.0 M	raya	0.4 M	xin	0.3 M
Albayan	1.5 M	levantine	0.3 M	al-watan	0.2 M
Hyt	1.5 M	ahram	0.3 M	asb	0.1 M
Aljaz	1.0 M				

Table 3.27: Number of contexts by Arabic text source for a CSLM training iteration

Several neural networks were trained using different projection sizes, 200, 220 and 250. The evolution of the perplexity on the training and the full development data set (dev06+dev07) with a network having a projection size of 200 is given in Table 3.28. The first few iterations result in a large decrease in perplexity of the training data, and to a lesser degree on the dev data. After eight or nine iterations, there is still a small gain on the training data, but this is not always the case for the dev data, for which the change in perplexity is seen to fluctuate.

<i>Iteration</i>	<i>Train</i>	<i>delta</i>	<i>Dev06+ Dev07</i>	<i>delta</i>	<i>Iteration</i>	<i>Train</i>	<i>delta</i>	<i>Dev06+ Dev07</i>	<i>delta</i>
1	364.8	-	791.2	-	7	172.4	4.9	648.7	9.0
2	236.0	128.8	717.9	73.2	8	168.5	4.0	647.9	0.8
3	208.0	28.0	686.6	31.4	9	165.0	3.5	640.1	7.8
4	193.5	14.4	671.3	15.3	10	162.2	2.8	640.7	-0.7
5	184.2	9.3	662.0	9.3	11	159.8	2.4	638.3	2.4
6	177.3	6.9	657.7	4.3	12	157.7	2.1	638.0	0.4

Table 3.28: Perplexity and change in perplexity of the Arabic training and development data as a function of training iteration for the CSLM

<i>Corpus</i>	<i>Size</i>	<i>Resampling Coeff. & Size</i>	
audio transcripts	17.3M	1.000	17.3M
giga xin	319.9M	0.006	1.92M
giga cna 2	247.7M	0.008	1.98M
agile bitex ce	178.0M	0.010	1.78M
VOARFA	35.4M	0.056	1.98M
Phoenixtv	88.0M	0.023	2.02M
ibm sina	283.3M	0.007	1.98M
bbn webdata	187.3M	0.010	1.87M

Table 3.29: The Mandarin text corpora used to train the continuous space language model, with the resampling coefficient and corpus sizes

The Mandarin continuous space model was trained on only a portion of the available text data as listed in Table 3.29. As for Arabic, all of the available transcripts of audio data (17.3M words) were used, as well as, other sources based on their recency and relevancy for a total of 1.3 billion words. The resampling coefficients of each individual source is also given in the table. It is 100% for the audio transcripts, and varies for the other sources in order to have on the order of two million words each at each iteration. These can be seen to range from 0.6% to 5% of the corpus size. Three neural networks were trained, having 200/250/300 hidden layers respectively. The corresponding perplexities on the combined development data set (dev07+eval07+dev08) are given in

Table 3.30. A standard 4-gram language model was formed by interpolating component LMs trained on 28 subcorpora. The perplexity on the combined dev07+eval07+dev08 data with this language model is 192.6. Interpolating this model (interpolation weight 0.484) with the three continuous space LMs using the weights shown in Table 3.30 reduces the perplexity of the development data to 171.8. This interpolated language model is what is referred to as the 4-gram CSLM.

A two pass decoding is used with PLP+MLP acoustic models (Fousek *et al.* 2008). The first pass generates a word lattice, followed by consensus decoding with 4-gram LM. This output is used for unsupervised acoustic model adaptation using CMLLR and MLLR, prior to a second decoding using the LM formed by interpolating the 4-gram LM with the 3 component CSLMs for the consensus decoding.

<i>Language Model</i>	<i>Hidden Layer Size</i>		
	200	250	300
CSLM	220.11	219.24	217.29
Interp. Weight	0.164	0.169	0.183

Table 3.30: Perplexity of the combined dev07+eval07+ dev08 data and interpolation weights of the 3 component CSLMs

Table 3.31 reports recognition error rates for Arabic and Mandarin on three data sets. For Arabic, a WER reduction of 0.5 to 0.7% is obtained using the CSLM. The gain in Mandarin CER is a bit smaller, 0.4 to 0.5%. Continuous space models have been trained for other languages (French, Spanish, Dutch, English) and tasks (Schwenk 2007). In all cases one or more CSLMs are interpolated with a standard 4-gram back off language model and improvements of the same order of magnitude have been consistently obtained.

<i>Arabic LM</i>	<i>eval06</i>	<i>dev07</i>	<i>eval07</i>
4-gram	19.1	12.1	13.7
CSLM	18.4	11.6	13.0
<i>Mandarin LM</i>	<i>dev07</i>	<i>evr07ns</i>	<i>dev08</i>
4-gram	11.3	10.3	10.5
CSLM	10.8	9.9	10.1

Table 3.31: Results comparing standard 4-gram backoff LM alone and interpolated with continuous space LMs on three data sets. Arabic WER (%) with automatic segmentation (top). Mandarin CER (%) with manual segmentation (bottom)

3.4.2.4 Experimental Evaluation in SMT

The continuous space LM was also used in a state-of-the-art phrase-based statistical machine translation (SMT) system. It is well known that the LM on the target language plays a crucial role in phrase-based MT system. It helps choosing among different possible translations of a word or phrase, decides whether reordering of the words should be performed and assures of course the smoothness of the produced sentence. Here we report on results with the CSLM achieved by the first author in an Arabic/English system that participated in the 2008 NIST MT evaluation (Schwenk and Esteve 2008).

The translation model was trained on all provided data (about 165M words) and the baseline 4-gram LM on all the texts of the English LDC Gigaword corpus (more than

3.5G words). In addition, a large collection of one billion 5-grams were used, obtained from texts collected by Google on the Internet. It has been reported that significant improvements of the performance of an SMT system can be obtained using all this data (Brants *et al.* 2007), but this requires substantial hardware equipment that exceeded by far our possibilities. Therefore, we applied various steps of filtering with the goal to keep only the most important n -grams (about 139M 4-grams). The results of this SMT system are summarized in Table 3.32. Adding this “Google LM” reduced the perplexity by about 9% in average, but most of the improvement is obtained on the WEB part of Dev06, which is actually not surprising given the source of this data. The CSLM achieved an additional perplexity reduction of almost 14% on top of the large LM which includes the filtered Google data. The storage requirements of the CSLM are also lower since it use a distributed representation of the knowledge.

System	Train	LM	Perplexity Dev06			BLEU Dev06			BLEU Eval08		
	#words	size	All	NW	Web	All	NW	Web	All	NW	Web
Gigaword LM	3.4G	3.7G	128.1	104.7	206.5	44.40	47.27	34.90	42.13		
+ Google LM	169M 4-grams	5.5G	114.5	99.0	161.7	44.70	47.22	36.11	41.90	47.20	33.50
CSLM	3.4G	≈ 1G	98.3	85.3	137.4	45.96	48.56	36.69	42.98	48.30	34.31

Table 3.32: Result summary for the baseline SMT system, adding parts of the Google n -grams and using the CSLM

The Google LM brought an improvement of 0.3 BLEU on the Dev data, all the improvements being again concentrated to the WEB part. Unfortunately, it turned out to be not useful on the Eval08 test data. The CSLM achieved an improvement of 1.1 BLEU on the test data, on top of this heavily optimized system. This final system achieved a very good ranking in the 2008 NIST MT evaluation. The official results of all participants have been published by NIST.⁹

3.4.2.5 Conclusion

Continuous space language models were initially developed in order to improve the generalization behavior when only a limited amount of in domain language model training data is available (Schwenk and Gauvain 2002). They were in particular used in the frame work of the DARPA EARS project to improve the modeling of conversational language. We have continuously improved the initial approach (Schwenk 2004; Schwenk and Gauvain 2005b; Schwenk 2007). Today, it is successfully used in all large vocabulary speech recognition systems developed at the LIMSI, consistently obtaining reductions of the word error rate of up to 1% absolute. We have developed very efficient algorithms to train the system on billions of words of text and to deploy it in real-time speech recognition system (Schwenk and Gauvain 2005a). In this work, we have reported results obtained in LIMSI’s Arabic and Mandarin speech recognition systems that were used in the DARPA GALE Program. Finally, the same technique was successfully ported to a large phrase-based Arabic/English statistical machine translation system.

⁹ <http://www.nist.gov/speech/tests/mt/2008/>

3.4.3. Rapid Unsupervised Topic Adaptation - A Latent Semantic Approach

Authors: Yik-Cheung Tam, Ian Lane and Tanja Schultz

3.4.3.1 Introduction

Topic adaptation is crucial for automatic speech recognition (ASR) and statistical machine translation (SMT) as it enables higher-level contextual information to be effectively incorporated into a background model to improve performance. Latent semantic analysis (LSA) is an effective approach to model topical information in an unsupervised fashion. Traditional latent semantic analysis is based on singular value decomposition (Deerwester *et al.* 1990; Bellegarda 2000; Kim 2004; Bellegarda 2005), while modern approaches are based on probabilistic modeling, including probabilistic LSA (Gildea and Hofmann 1999; Mrva and Woodland 2004; Akita and Kawahara 2004), and latent Dirichlet allocation (LDA) (Blei *et al.* 2003). LDA has drawn significant attention in the ASR community, particularly on language model adaptation (Tam and Schultz 2005; Tam and Schultz 2006; Hsu and Glass 2006; Heidel *et al.* 2007; Chien and Chueh 2008; Liu and Liu 2008). A related approach also includes topic mixtures at a sentence level (Iyer and Ostendorf 1999).

We propose a unified topic adaptation framework for monolingual and cross-lingual adaptation based on LDA. We have made several extensions and contributions to LDA within the adaptation framework. For monolingual modeling, our contributions are multi-folded: we generalize LDA to latent Dirichlet-Tree allocation to model topic correlation via a tree-based topic prior; we extend latent Dirichlet-Tree allocation to its N -gram version to relax the “bag-of-words” assumption that word ordering is not modeled in LDA; we formulate a fractional Kneser-Ney smoothing for N -gram LSA to manage fractional counts; we derive a variational Expectation-Maximization (EM) algorithm for efficient model training on large data. For cross-lingual modeling, we present a bilingual LSA approach which enables using a source text of one language to adapt the models of another language. Not only our framework adapts a language model (LM) monolingually for ASR, but it also adapts language models of a target language, translation lexicons and phrase tables cross-lingually for SMT.

3.4.3.2 Latent Semantic Analysis

The goal of latent semantic analysis is to find a set of latent patterns to describe a data set in an unsupervised fashion. The idea is similar to dimensionality reduction such as principal component analysis (Duda *et al.* 2001) to find a subset of eigenvectors. Latent semantic indexing (LSI) is a traditional vector-space approach for LSA via applying singular value decomposition (SVD) on a term-document matrix of document corpora. The drawback of a vector-space approach is the difficulty to incorporate prior knowledge. On the other hand, probabilistic modeling is a principled way to incorporate prior knowledge, such as latent Dirichlet allocation (LDA) for latent semantic analysis. For background details, we first review LDA and then describe our extension, latent Dirichlet-Tree allocation, to model topic correlation. We present N -gram latent Dirichlet-Tree allocation and efficient methods for model training and smoothing.

3.4.3.2.1. Latent Dirichlet Allocation

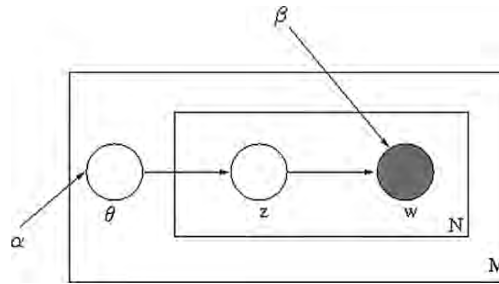


Figure 3.12: Graphical representation of document generation using latent Dirichlet allocation. N and M denote number of words in a document and number of documents in training corpora respectively

One distinctive feature in latent Dirichlet allocation over a SVD-based approach is the use of a Dirichlet prior over a topic distribution θ as illustrated in Figure 3.12. The figure describes a generative procedure for a document w_1^N in LDA as follows:

1. Sample θ from $\text{Dirichlet}(\cdot, \{\alpha_k\})$.
2. For each word w_i in a document w_1^N ,
 - Sample a latent topic index z_i from $\text{Multinomial}(\theta)$.
 - Sample w_i from $p(w|z_i)$ (or β_{wz_i}).

The model parameters Λ of LDA are $\{\alpha_k\}$ for a Dirichlet prior, and $\{p(w|k)\}$ for a set of topic-dependent unigram language models with k and w denoting a topic index and a term index respectively. Intuitively, α_k can be interpreted as a prior pseudo count for topic k .

3.4.3.2.2. Latent Dirichlet-Tree Allocation

One assumption in latent Dirichlet allocation is the use of a Dirichlet prior implying topic independence. In other words, knowing the proportion of one topic does not provide any information about the proportion of another topic. In reality, this assumption may not be true since topics can be correlated. For instance, news articles in a newspaper website are usually organized into a topic hierarchy to represent topic correlation. This motivates the extension of latent Dirichlet allocation into latent Dirichlet-Tree allocation (Tam and Schultz 2007a) to capture topic correlation via a structural Dirichlet-Tree prior. A Dirichlet prior employed in LDA is a special case of a Dirichlet-Tree prior since a Dirichlet prior can be visualized as a flat tree with depth one. Sampling a Dirichlet distribution becomes labeling tree branches from a node with probabilities summing to unity. In general, a Dirichlet-Tree can have different depths and structures. Figure 3.13 illustrates a depth-two Dirichlet-Tree where a root node is represented by a Dirichlet distribution with multiple branches and Dirichlet nodes at the bottom allow binary branches only.

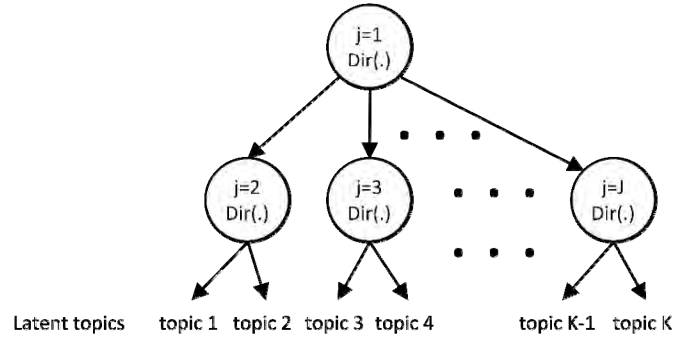


Figure 3.13: A Dirichlet-Tree prior of depth two.

Figure 3.14 illustrates the effect of the tree structure in terms of training likelihood by varying the number of branches in a Dirichlet node. Results show that the training convergence is optimal when a balanced binary tree is employed. As the number of branches increases, the independence assumption among topics becomes stronger, and thus, slowing down the training convergence. In an extreme case in latent Dirichlet allocation, the training convergence is the slowest.

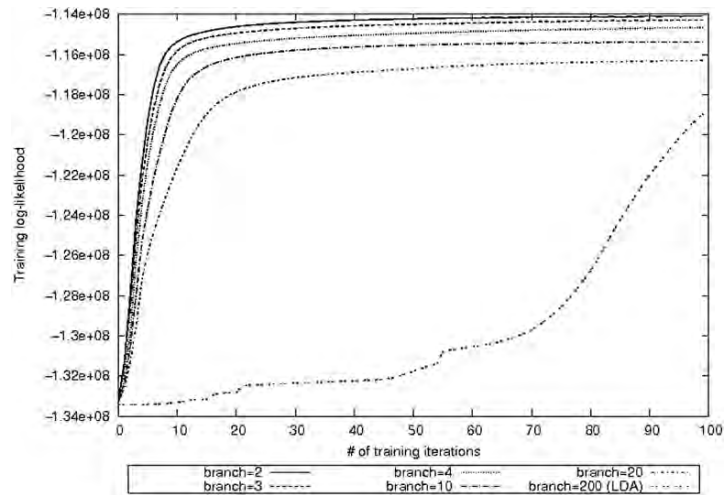


Figure 3.14: Training log-likelihood of latent Dirichlet-Tree allocation with different number of branches in a Dirichlet node using Xinhua News 2002 corpora (13M words).

It seems that latent Dirichlet-Tree allocation is more complex than latent Dirichlet allocation. However, both models have the same graphical model representation, and thus, share a similar document generation procedure. The major difference lies in the sampling procedure for a topic distribution θ :

1. Sample a vector of branch probabilities $b_j \sim \text{Dirichlet}(\cdot, \{\alpha_{j_c}\})$ for each node $j = 1 \dots J$ where $\{\alpha_{j_c}\}$ denotes the parameter of a Dirichlet distribution at node j , i.e. a pseudo-count of an outgoing branch c at node j .

2. Compute a topic distribution as follows:

$$\theta_k = \prod_{jc} b_{jc}^{\delta_{jc}(k)} \quad (3.30)$$

where $\delta_{jc}(k)$ is an indicator function, which sets to unity when the c^{th} branch of the j^{th} node leads to the leaf node of topic k , and zero otherwise. The k^{th} topic weight θ_k is computed as a product of sampled branch probabilities from a root node to a leaf node corresponding to topic k .

3. For each word w_i in a document w_1^N ,
- Sample a latent topic index z_i from $\text{Multinomial}(\theta)$.
 - Sample w_i from $p(w|z_i)$

Our model training employs variational EM similar to latent Dirichlet allocation except a difference in the variational E-step to handle a tree-based prior in Equation (3.30). The difference can be understood via the conjugate property as follows: A Dirichlet-Tree posterior has the same form as a Dirichlet-Tree prior given a topic sequence z_1^N since:

$$\begin{aligned} p(b_1^J | z_1^N) &\propto p(z_1^N | b_1^J) \cdot p(b_1^J; \{\alpha_{jc}\}) \propto \left(\prod_{i=1}^N \prod_{jc} b_{jc}^{\delta_{jc}(z_i)} \right) \cdot \prod_{jc} b_{jc}^{\alpha_{jc}-1} \\ &= \prod_{jc} b_{jc}^{(\alpha_{jc} + \sum_{i=1}^N \delta_{jc}(z_i)) - 1} \end{aligned}$$

This result suggests that the posterior branch count γ_{jc} can be computed by accumulating the expected branch counts from current observations:

E-Steps:

$$\gamma_{jc} = \alpha_{jc} + \sum_{i=1}^N E[\delta_{jc}(z_i)] \quad (3.31)$$

$$= \alpha_{jc} + \sum_{i=1}^N \sum_{k=1}^K q_{ik} \cdot [\delta_{jc}(k)] \quad (3.32)$$

$$q_{ik} \propto p(w_i | k) \cdot e^{E_q[\log \theta_k; (\gamma_{jc})]} \quad (3.33)$$

where $E_q[\log \theta_k] = \sum_{jc} \delta_{jc}(k) E_q[\log b_{jc}] = \sum_{jc} \delta_{jc}(k) (\Psi(\gamma_{jc}) - \Psi(\sum_c \gamma_{jc}))$ where q_{ik} denotes $q(z_i = k | w_1^N)$ meaning the variational topic posterior of word w_i . $\Psi(\cdot)$ denotes the derivative of the logarithm of the Gamma function. Equations (3.31) and (3.33) are executed iteratively until convergence is reached. Equation (3.31) can be implemented as propagation and summation of fractional topic counts q_{ik} from the leaf nodes to the root node in a bottom-up fashion as shown in Figure 3.15.

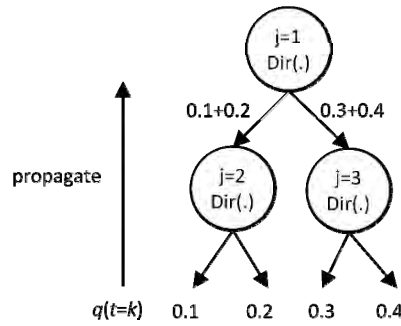


Figure 3.15: Variational E-step as bottom-up propagation and summation of fractional topic counts.

M-Step:

$$\hat{p}(w|k) = \frac{C_k(w)}{\sum_{k=1}^K C_k(w)} \tag{3.34}$$

Where $C_k(w) = \sum_{i=1}^N q_{ik} \cdot \delta(w_i, w)$

The re-estimation formula for $(p(w|k))$ is the weighted relative frequency in Equation (3.34) where $\delta(w_i, w)$ denotes a Kronecker Delta function. The $\{\alpha_{jc}\}$ parameters can be re-estimated with an iterative approach such as Newton-Raphson or simple gradient ascent.

3.4.3.2.3. N-gram Latent Dirichlet-Tree Allocation

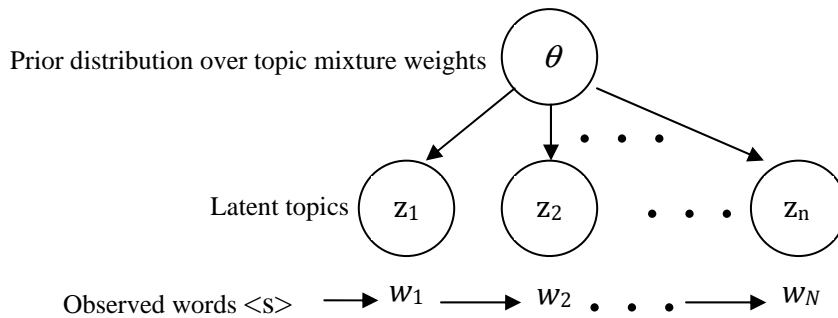


Figure 3.16: Graphical representation of bigram LSA. Adjacent words of a document are connected together to form a Markov chain from left to right.

Latent Dirichlet allocation makes a “bag-of-word” assumption that each word is generated irrespective of its position in a document. To relax this assumption, bigram LSA has been proposed (Wallach 2006) that modifies the graphical structure of LDA.

Figure 3.16 shows the graphical representation of bigram LSA where adjacent words of a document are connected together to form a Markov chain. The representation can be extended easily to its N -gram version. With the modified graphical structure, the document generation of bigram LSA is similar to LDA except that a previous word is taken into consideration to generate a word:

4. Sample θ from $\text{Dirichlet}(\theta, \{b_{jc}\})$.
5. For each word w_i in a document w_1^N ,
 - Sample a latent topic index $z_i \sim \text{Multinomial}(\theta)$.
 - Sample w_i given the previous word w_{i-1} and the topic label $z_i: w_i \sim p(\cdot | w_{i-1}, z_i)$,

Instead of using Gibbs sampling (Wallach 2006) for inference, we estimate bigram LSA using variational EM. It can be shown that the E-steps are slight modification of LSA via replacing the k^{th} unigram LM $p(w_i|k)$ in Equation (3.34) with the k^{th} bigram LM $p(w_i|w_{i-1}, k)$. For practical reason, we approximate $p(w_i|w_{i-1}, k)$ with $p(w_i|k)$ to avoid keeping all topic-dependent bigram language models into memory in the E-steps. In the M-step, the accumulated topic-dependent bigram counts $C_k(u, v)$ are used to re-estimate a new model. With maximum likelihood principle, the estimated model is an unsmoothed bigram LSA based on relative bigram counts $C_k(u, v)$ that assigns zero probabilities to unseen bigrams undesirably. In the following section, we investigate fractional Kneser-Ney smoothing (Bisani and Ney 2008; Tam and Schultz 2008) to handle unseen bigrams and deal with fractional bigram counts.

Fractional Kneser-Ney Smoothing: The state-of-the-art smoothing algorithm for back off language models is based on Kneser-Ney smoothing (Kneser and Ney 1995). The belief of its success is due to the preservation of marginal distributions. However, the original formulation is defined on integral counts which is not suitable for N -gram LSA using fractional counts. Therefore, we have proposed fractional Kneser-Ney smoothing as a generalization of the original formulation.

The state-of-the-art smoothing algorithm for backoff language models is based on Kneser-Ney smoothing (Kneser and Ney 1995). The belief of its success is due to the preservation of marginal distributions. However, the original formulation is defined on integral counts which is not suitable for N -gram LSA using fractional counts. Therefore, we have proposed fractional Kneser-Ney smoothing as a generalization of the original formulation.

The interpolated form of a bigram LM with the absolute discounting can be expressed as follows:

$$P_{KN}(v|u) = \frac{\max(C(u, v) - D, 0)}{C(u)} + \lambda(u)P_{KN}(v)$$

where $D \in (0, 1)$ is a discounting factor and $P_{KN}(v)$ is treated as a model parameter to

make the above equality holds. Intuitively, the solution for $P_{KN}(v)$ can be interpreted as the relative frequency over discounts (or heldout counts) as follows:

$$P_{KN}(v) = \frac{C_{\leq D}(\cdot, v) + D \cdot N_{> D}(\cdot, v)}{\sum_w C_{\leq D}(\cdot, w) + D \cdot N_{> D}(\cdot, w)} \quad (3.35)$$

where $C_{\leq D}(\cdot, v)$ denotes the sum of bigram counts preceding v and smaller than D , while $N_{> D}(\cdot, v)$ denotes the number of word types preceding v with bigram counts bigger than D . Equation (3.35) generalizes the original formulation since each integral count is always bigger than D by definition. Therefore, the term $C_{\leq D}(\cdot, v)$ vanishes thus recovering the original formulation. In general, the approach can apply to a backoff N -gram LM or a factored language model via propagating heldout N -gram counts from a higher-order LM to a lower-order LM recursively.

3.4.3.2.4. Two-Stage Unsupervised LM Adaptation

Given word hypotheses from the previous decoding pass as adaptation text, we apply E-steps (Equations (3.31) – (3.33)) to estimate variational Dirichlet posterior of each node in a Dirichlet-Tree. We estimate the topic weights of the adaptation text using the following equation:

$$\theta_k \propto \prod_{jc} \left(\frac{\gamma_{jc}}{\sum_{c'} \gamma_{jc'}} \right)^{\delta_{jc}(k)} \quad (3.36)$$

Then we apply the topic weights to obtain adapted unigram, bigram and trigram LSA:

$$\begin{aligned} p_a(w) &= \sum_{k=1}^K p(w|k) \cdot \hat{\theta}_k \\ p_a(w|v) &= \sum_{k=1}^K p(w|v, k) \cdot \hat{\theta}_k \\ p_a(w|u, v) &= \sum_{k=1}^K p(w|u, v, k) \cdot \hat{\theta}_k \end{aligned} \quad (3.37)$$

At the first stage of LM adaptation, we integrate the LSA marginals into a background LM via marginal adaptation (Kneser *et al.* 1997):

$$p_a^{(1)}(w|h) = \frac{\alpha(w) \cdot p_{bg}(w|h)}{\sum_v \alpha(v) \cdot p_{bg}(v|h)} \quad (3.38)$$

Where $\alpha(w) = \left(\frac{p_a(w)}{p_{bg}(w)} \right)^\epsilon$ and ϵ is a tuning parameter between 0 and 1. Intuitively, the likelihood ratio $\alpha(w)$ controls when to boost/de-emphasize the probability of an on-topic/off-topic N -gram entry. Computing the normalization factor in Equation (3.38) involves summing over all possible words v which is prohibitive on large vocabulary. However, an efficient normalization is described by Kneser *et al.* (1997) for a back-off

LM. At the second stage of LM adaptation, the adapted LM from the first stage of adaptation is linearly interpolated with the adapted bigram and trigram LSA as follows:

$$p_a^{(2)}(w|h) = \lambda_1 \cdot p_a^{(1)}(w|h) + \lambda_2 \cdot p_a(w|v) + \lambda_3 \cdot p_a(w|u, v)$$

where $\{\lambda_i\}$ are interpolation weights tuned using a development set.

When v is a stopword such as auxiliary verbs, articles, conjunctions, sentence boundary markers and punctuations, we do not adapt their N -gram probabilities because predicting stopwords mostly relies on the syntactic context, but not the topical context.

3.4.3.3 Bilingual Latent Semantic Analysis

The goal of cross-lingual adaptation is to use information of one language to adapt the models of another language. Bilingual LSA (bLSA) (Tam *et al.* 2007b) provides a platform for cross-lingual topic adaptation using the fact that topical information are language independent, and thus, transferable among languages. To implement this property, bilingual LSA enforces a one-to-one topic correspondence between source and target LSA models so that an inferred topic distribution can be transferred from a source language to a target language. For parallel document pairs that are faithful translations, a one-to-one topic correspondence is reasonable. Figure 3.17 illustrates the idea of topic transfer between mono-lingual LSA models followed by language model, translation lexicon and phrase table adaptation in a speech translation system. Since our approach adapts models before translation using a source text, an extra decoding pass to obtain an initial translation output for monolingual adaptation is not needed. For the same reason, propagation of translation errors can be avoided using a source text for adaptation.

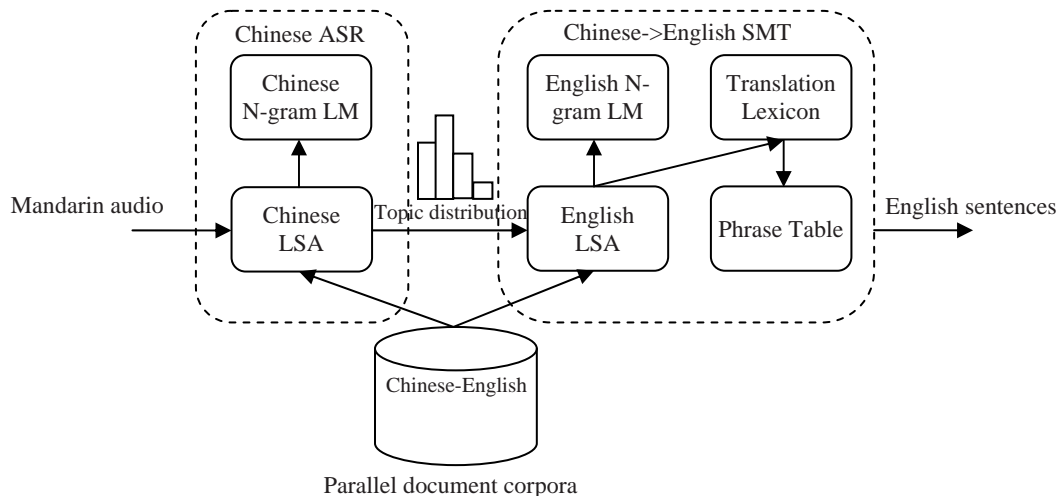


Figure 3.17: Bilingual LSA-based adaptation via transfer of topic distribution from a source language to a target language for speech translation.

In the following sections, we present an approach for bilingual LSA training. In addition, we borrow the LM adaptation techniques from a monolingual setting to a cross-lingual setting.

3.4.3.3.1. Bilingual LSA Training

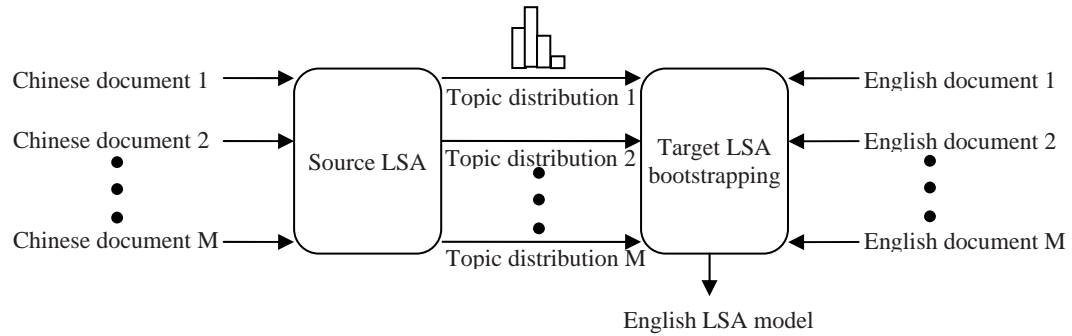


Figure 3.18: LSA bootstrapping via sharing of variational Dirichlet posteriors for parallel documents.

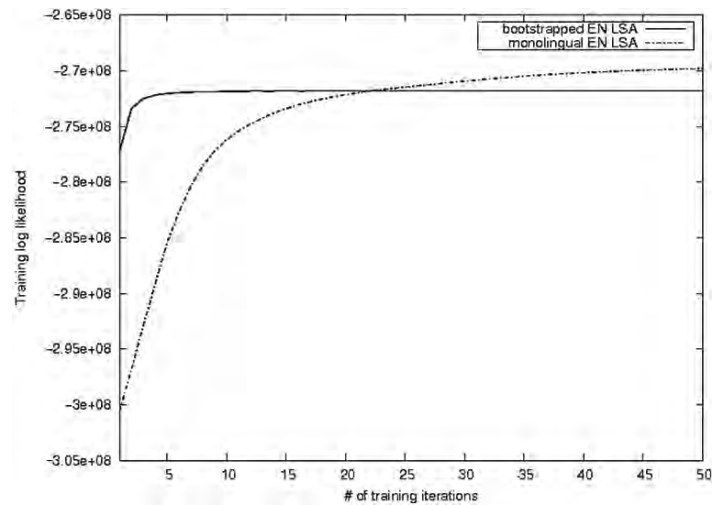


Figure 3.19: Training log likelihood of a bootstrapped English LSA from a Chinese LSA compared to a flat monolingual English LSA.

Figure 3.18 illustrates the idea of enforcing a one-to-one topic correspondence of parallel document pairs to bootstrap target LSA from source LSA denoted as $bLSA_{(src,tgt)}$. Since the topic transfer can be bidirectional, we can perform bilingual LSA training in a reverse manner, i.e., training a target LSA model followed by bootstrapping

a source LSA model denoted as $bLSA_{(tgt,src)}$. However, previous experiments have confirmed that the bootstrapping directions make no difference in cross-lingual LM adaptation performance (Tam *et al.* 2007b).

3.4.3.3.2. Cross-Lingual LM Adaptation

Marginal language model adaptation in a cross-lingual setting can be performed in almost the same manner as in a monolingual setting as described in Section 3.4.3.2.4. In the cross-lingual case, a source text is used for adaptation. First, we estimate the topic weights using source LSA and the source text using Equation (3.36). Then we apply the inferred source topic weights ($\hat{\theta}_k^{(CH)}$) into target LSA to obtain in-domain LSA marginals as follows:

$$P_{EN}(w) = \sum_{k=1}^K p^{(EN)}(w|k) \cdot \hat{\theta}_k^{(CH)} \quad (3.39)$$

Finally, we follow the same marginal LM adaptation procedure described in Section

3.4.3.3.3. Translation Lexicon Adaptation

Not only bilingual LSA applies to LM adaptation, but it also extends to translation lexicon adaptation (Tam and Schultz 2007b). It is motivated from an observation that a source word can be translated into different target words depending on a topical context. A popular example is “bank”, which can be related to either a “financial bank” or a “river bank”. An adapted translation lexicon can rescore phrase pairs of a phrase table to reflect their topical dependency towards a source text.

Motivated by information theory, we formulate the adaptation problem as marginal adaptation. The goal is to minimize the Kullback-Leibler divergence between an adapted lexicon $p_a(c|e)$ and a background lexicon $p_{bg}(c|e)$ such that the lexical marginals computed from the adapted lexicon are equal to in-domain marginals $p(c|d_{ch})$ that are estimated from a source text. Thus, an objective function is defined as follows:

Minimize $\sum_e p_a(e) \cdot KL(p_a(\cdot|e) || p_{bg}(\cdot|e))$ such that $\forall c: \sum_e p_a(e) \cdot p_a(c|e) = p(c|d_{ch})$, $\forall e: \sum_c p_a(c|e) = 1$
 where d_{ch} is a source text to estimate a unigram distribution. The optimal solution is as follows:

$$p_a(c|e) \propto p_{bg}(c|e) \cdot e^{\lambda_c} \propto p_{bg}(c|e) \cdot e^{\sum_j \lambda_j f_j(c,e)}$$

where $f_j(c, e) = \begin{cases} 1 & \text{if } c = j \\ 0 & \text{otherwise} \end{cases}$

$f_j(c, e)$ is a unigram feature function independent of e . Since the solution of an adapted lexicon is in an exponential form, the optimization problem is similar to the maximum entropy setting. Therefore, we solve λ_j using generalized iterative scaling (GIS) (Darroch and Ratcliff 1972) as follows:

$$\lambda_j^{(t+1)} = \lambda_j^{(t)} + \log \frac{\bar{E}[f_j(c,e)]}{E[f_j(c,e)]} \approx \lambda_j^{(t)} + \log \frac{p(c=j|d_{ch})}{\sum_e p_a^{(t)}(c=j|e) \cdot p_{blsa}(e|d_{ch})} \quad (3.40)$$

where t denotes an index of a GIS iteration with $p_a^{(0)}(c|e) = p_{bg}(c|e)$ $\lambda_j^{(0)} = 0$. $p_a(e)$ is approximated by the English LSA marginals $P_{blsa}(e|d_{ch})$ computed from bilingual LSA. Since the range of e in Equation (3.40) is limited to the number of possible translation pairs (c, e) in a lexicon, computing the denominator term is efficient without evaluating all possible e . We estimate $p(c|d_{ch})$ using the smoothed relative word frequency of a source text with Good-Turing discounting scheme. Since the optimization is convex, a global optimal solution of an adapted lexicon is guaranteed.

3.4.3.3.4. Phrase Table Adaptation

Ideally, an adapted translation lexicon can be applied directly to phrase extraction. But, this involves an extra implementation work into the phrase extraction algorithm. A simple approach is to take a background phrase table and assume that good phrase pairs are already captured by the table. With an adapted translation lexicon, we can score each phrase pair (c_1^I, e_1^J) in the background phrase table as follows:

$$p_a(c_1^I|e_1^J) = \prod_{i=1}^I \frac{1}{J_i} \cdot \sum_j^{J_i} p_a(c_i|e_j) \quad (3.41)$$

where $0 < J_i < J$ denotes the effective number of target word e_j aligned to a source word c_i after pruning unlikely lexical entry with probability less than 10^{-4} in the adapted translation lexicon. The motivation is to have a “sharper” probability average to make the phrase score more discriminative. The NULL model $p(c|NULL)$ or the minimum of $p(c|NULL)$ is used as a backoff model to avoid zero probability of an unseen word translation. $p_a(c_1^I|e_1^J)$ can be defined in the same manner. For phrase table adaptation, these two bLSA-adapted phrase scores are simply appended to the background phrase table for subsequent minimum error rate training (MERT) and SMT decoding.

3.4.3.4 Experimental Setup

The proposed topic adaptation framework was evaluated monolingually for ASR and cross-lingually for SMT.

3.4.3.4.1. ASR Setup

Our CMU-InterACT GALE-Phase 3 transcription systems were employed for Chinese Mandarin and Arabic evaluation. Chinese and Arabic are very different linguistically. Chinese has a compact vocabulary formed via a combination of Chinese characters while Arabic has a large vocabulary due to morphology. The sizes of Chinese and Arabic vocabulary are 108k and 737k respectively in our GALE Systems. Both systems were trained with the public LDC data released from the GALE Program. The corpora statistics for acoustic and LM training are shown in Table 3.33. The development and test sets are shown in Table 3.34. Dev08 and Dev07 were employed as the tuning set for Mandarin and Arabic system respectively while the remaining sets were test sets. Please refer to Section 3.5.3 and Section 3.6.4 for detailed descriptions of our Mandarin and Arabic systems, respectively. The number of latent topics K in LSA was set to 200 based on our past knowledge of language model adaptation for ASR. A balanced binary Dirichlet-Tree prior was used. A two-stage LM adaptation was performed as described in Section 3.4.3.2.4 using the word hypotheses from the final decoding pass. Incremental LSA and N -gram LSA based lattice rescoring were then applied to obtain the final word hypotheses. Character error rate (CER) and word error rate (WER) were reported for Mandarin and Arabic evaluation respectively. Matched pairs sentence-segment word error test (Gillick and Cox 1989) was performed for significance test using the NIST scoring tool.

	Mandarin		Arabic	
	AM	LM	AM	LM
GALE-Phase 3	1300hr	1.5B	1500hr	962M

Table 3.33: Size of acoustic model (AM) and language model (LM) training corpora for GALE Phase 3 Mandarin and Arabic transcription systems in terms of number of hours and word tokens.

	Mandarin			Arabic		
	Dev08	Eval07u	Eval07r	Dev07	Dev08	Eval07u
	0.98hr	1.15hr	1.64hr	2.58hr	3.04hr	3.4hr

Table 3.34: Statistics of development and test sets for transcription evaluation. “Eval07u” and “Eval07r” stand for unsequestered and retest portions of Eval07 respectively.

3.4.3.4.2. MT Setup

SMT System	LM training	Parallel training corpora	
	English LM	Chinese	English
Development	500M	59M	67M
GALE-Phase 2	2.7B	232M	260M

Table 3.35: Size of language model training corpora and parallel training corpora for phrase extraction in terms of number of word tokens.

Our medium-scale SMT development system and the large-scale GALE Phase 2 SMT system were employed that translate from Chinese to English (Hildebrand *et al.* 2008). Phrase extraction was performed using parallel corpora as shown in Table 3.35. The table also shows that 4-gram and 5-gram LM were trained using 500M and 2.7B English words for the medium-scale and the GALE Systems, respectively.

The SRILM toolkit (Stolcke 2002) was used for language model training with the modified Kneser-Ney smoothing. SMT was performed by constructing a translation lattice, which contained all possible matched bilingual phrase pairs of an input source sentence. In text translation using the GALE Phase 2 system, part-of-speech based word reordering (Rottmann and Vogel 2007) was performed on an input sentence to produce an input source lattice before building the translation lattice. Search was then performed on this lattice using our STTK beam search decoder (Vogel *et al.* 2003). The word reordering window was set to three for the medium-scale system, while monotonic decoding was applied for the GALE System since word reordering was already applied in the source lattices. An optimal path was returned with the maximum translation score consisting of a log-linear combination of feature functions, such as, language model probability, distortion penalty, word-count penalty, phrase count, phrase-alignment scores and bilingual LSA-adapted phrase scores. The corresponding feature weights were optimized in BLEU (Papineni *et al.* 2002) using minimum error rate training (Och 2003). Table 3.36 shows the development and test sets. All systems were tuned on MT03. The test set for text translation was MT06 with four English references. For speech translation, the test set was the unsequestered broadcast news portion of Mandarin Eval07 with confusion-network-like English references to represent multiple translation options of an English phrase. The bilingual LSA model was trained using the Chinese–English parallel document corpora with statistics shown in Table 3.35. Our bilingual LSA training did not take advantage of the large parallel corpora in phrase extraction due to the loss of document boundary. However, encouraging results were still achieved. The source and target vocabulary in bilingual LSA were limited to words occurred in the phrase table. The Stanford Chinese word segmenter (Tseng *et al.* 2005) was applied to segment the Chinese side of the parallel corpora. Monolingual LSA training was first applied on the Chinese side followed by LSA bootstrapping on the English side.

Set	Sentence	Document	Reference
MT03 (dev)	919	100	4
MT06	1664	79	4
Eval07u.BN	314	32	CN

Table 3.36: Statistics of a development set (MT03), a test set (MT06) for text translation, and the unsequestered broadcast news (BN) portion of Eval07 for speech translation. Confusion network (CN) is used to represent multiple translation options of a target phrase in Eval07 test set.

Each source test document was used for cross-lingual adaptation before SMT decoding. First, marginal adaptation was applied to an English background language model. Words on the stop-word list plus punctuation were filtered out from language model adaptation. Secondly, translation lexicons were also adapted via marginal adaptation. Bilingual LSA-adapted phrase scores on both translation directions were added as new feature functions to the background phrase table in an offline fashion before MERT and SMT decoding. Thirdly, N -gram LSA was served as an additional language model feature that is topic adaptive to each source test document. The MT performance was reported using BLEU and NIST (Doddington 2002) scores. Significance testing was performed using a bootstrapping approach described by Zhang and Vogel (2004).

For speech translation, we evaluated the effectiveness of topic adaptation using different source inputs on our GALE Phase 2 Mandarin-English SMT system without part-of-speech reordering feature. We translated word hypotheses from unadapted and LSA-adapted GALE Phase 3 Mandarin transcription systems with character error rates 5.6% and 5.2% respectively on the unsequestered broadcast news portion of Eval07 test set. We also translated manual transcription to serve as an upper bound performance for comparison.

3.4.3.4.3. Recognition Results

The upper section of Table 3.37 columns 2-4 shows the overall LM adaptation results on Mandarin before cross adaptation. To illustrate the effect of LSA during decoding, the first background setting is compared to the second background setting that enables LSA during decoding. Both settings are then followed by lattice rescoring using LSA and bigram LSA (biLSA). Applying LSA during decoding helps on all test sets compared to the unadapted baseline. In addition, applying LSA for lattice rescoring yields further CER reduction. Bigram-LSA rescoring yields an additional 2.9 - 4.3% relative CER reduction compared to LSA rescoring, which is statistically significant at a $\leq 5\%$ significance level in all test cases. Combining bigram and trigram LSA gives a slight gain on Eval07r. Comparing to the unadapted baseline, the final relative reduction after LM adaptation is between 5% - 7.6%, which is statistically significant at a 0.1% significance level in all test cases. The results after cross adaptation follows a similar trend with 3.3% - 6.7% relative CER reduction compared to the unadapted baseline.

Mandarin BN (BC)	Dev08	Eval07u	Eval07r	Dev08		Eval07u	
Background	11.6	14.0	11.8	5.4	(17.8)	5.6	(24.8)
+LSA	11.4	13.9	11.6	5.0	(17.7)	5.6	(24.7)
+biLSA (Kneser)	11.0*	13.6*	11.1*	5.0	(17.0*)	5.4	(24.3)
background (LSA)	11.5	13.8	11.7	5.5	(17.4)	5.7	(24.3)
+LSA	11.2	13.7	11.5	5.2	(17.1)	5.5	(24.2)
+biLSA (Witten)	10.9*	13.3*	11.1*	5.0	(16.6*)	5.2*	(23.9)
+biLSA (Kneser)	10.8*	13.3*	11.0*	4.9*	(16.7)	5.2*	(23.8*)
+triLSA (Kneser)	10.8*	13.3*	11.1*	5.1	(16.3*)	5.2*	(23.9)
+allLSA (Kneser)	10.8*	13.3*	10.9*	5.0	(16.5*)	5.2*	(23.9)
Cross-adaptation with IBM							
background	9.0	10.8	9.0	4.2	(13.7)	4.1	(19.6)
+LSA	8.9	10.6	8.7	4.1	(13.5)	3.9	(19.4)
+biLSA	8.6*	10.4	8.5*	3.9*	(13.3)	3.6*	(19.3)
background (LSA)	9.0	10.6	8.8	4.2	(13.7)	3.9	(19.4)
+LSA	8.8	10.5	8.6	4.1	(13.6)	3.8	(19.2)
+biLSA	8.7	10.3*	8.4*	3.8*	(13.5)	3.5*	(19.1)

Table 3.37: Lattice rescoring results on overall (columns 2 - 4) and BN (BC) in character error rate using the CMU-InterACT Mandarin transcription system for the GALE Phase 3 evaluation. * denotes results that are significantly better than LSA at $\leq 5\%$ level of significance.

The performance trend is similar on Arabic as shown in Table 3.38 columns 2 - 4. LSA gives slight WER reduction compared to the unadapted baseline. Moreover, bigram

LSA achieves additional 3% relative WER reduction compared to LSA on the unseen Dev08, which is statistically significant at a 0.1% significance level.

Arabic	Dev07	Dev08	Eval07u	Dev07	Eval07u
background	14.3%	16.4	22.7	11.6 (19.4)	20.8 (24.7)
+LSA	14.2	16.4	22.7	11.5 (19.2)	20.6 (24.8)
+biLSA (Witten)	13.9	15.9*	22.4*	11.0* (19.0)	20.4* (24.6)
+biLSA (Kneser)	13.8*	15.9*	22.5	11.0* (18.9)	20.4* (24.7)
Cross-adaptation with IBM					
background	11.8	13.9	20.3	9.9 (15.1)	18.7 (22.0)
+LSA	11.8	13.8	20.3	9.8 (15.5)	18.6 (21.9)
+biLSA (Kneser)	11.7	13.6*	20.1	9.8 (15.2)	18.2* (22.1)

Table 3.38: Lattice rescoring results on overall (columns 2 - 4) and BN (BC) in word error rate using the CMU-InterACT Arabic transcription system for the GALE P3 evaluation. * denotes results that are significantly better than LSA at $\leq 5\%$ level of significance.

The performance breakdown in broadcast news (BN) and broadcast conversation (BC) are shown in Table 3.37 and 3.38 columns 5 - 8 for Mandarin and Arabic, respectively. Bigram LSA generally works better on BN than BC in terms of relative error reduction although the recognition accuracy is much better on BN than BC. One possible reason is the mismatch of the LM training data that are mainly newspaper text closer to BN than BC.

Not only fractional Kneser-Ney smoothing is comparable to Witten-Bell smoothing in performance, but the model is also more compact. For instance, the size of the compressed Arabic model after fractional Kneser-Ney smoothing is 2.4Gb, while the model size after Witten-Bell smoothing is 3.7Gb yielding 35% relative reduction in model size.

3.4.3.4.4. Text Translation Results

English model	EN 4-gram (500M)		EN 5-gram (2.7B)	
	BLEU	NIST	BLEU	NIST
Baseline	28.06%	8.71	31.49	9.23
bLSA-adapted	28.62	8.80	31.94	9.31
bLSA-adapted lex	28.59	8.92*	32.03	9.34
bLSA-adapted + lex	28.91*	8.97*	32.09	9.37
mono LSA-adapted	28.41	8.81	-	-
mono LSA-adapted lex	28.72	8.96*	-	-
mono LSA-adapted + lex	28.97*	9.00*	-	-
bLSA-adapted + biLSA	29.08*	8.99*	-	-

Table 3.39: MT06 evaluation results on BLEU and NIST using 67M-word (260M-word) parallel corpora for phrase extraction and 500M-word (2.7B-word) English corpora for language model training. Vocabulary size of a target language model is 1.3M (4.1M). Four English references are used for scoring. * denotes results that are significantly better than the unadapted baseline at a 95% confidence interval.

The upper section of Table 3.39 shows the translation performance in BLEU and NIST on MT06 using the medium-scale SMT system. 2% relative improvement in BLEU is achieved compared to the unadapted baseline after applying bilingual LSA-based

language model adaptation and translation lexicon adaptation separately. When both techniques are applied simultaneously, the gain is additive giving 3% relative improvement in BLEU compared to the unadapted baseline. The improvement is statistically significant at the 95% confidence interval [27.29%, 28.84%] with respect to the unadapted baseline. The same performance trend in NIST is also observed with 3% relative improvement compared to the unadapted baseline. The improvement is statistically significant at the 95% confidence interval [8.61, 8.85] with respect to the unadapted baseline.

The middle section of Table 3.39 shows that mono-lingual LSA adaptation using the first-pass translated hypotheses achieves similar performance compared to bilingual LSA adaptation using the source text. In other words, the source text and the initial MT hypotheses are equally effective for LSA adaptation since LSA is robust against translation errors in the adaptation text. We conjecture that the quality of translation of topical unigrams should be acceptable in the initial translation. However, in terms of computation, bilingual LSA is more elegant and requires only a single decoding pass compared to monolingual LSA.

Bigram LSA and trigram LSA show improvement in translation performance in addition to the bLSA-adapted language model. However, slight degradation occurs after adding the adapted translation lexicons.

Table 3.40 shows some sample sentences demonstrating some degree of semantic paraphrasing with bilingual LSA, such as *people of Denmark* versus *Danish people*, and *told* versus *sighed*.

Sample output 1	
Baseline	To achieve the extensive support from the international community to save this problem, the government of Denmark, and Denmark is very important.
Bilingual LSA	To achieve the extensive support from the international community to save this problem, the Danish government and people of Denmark is very important.
Reference	It is extremely important to the Danish government and the Danish people to obtain the broad support of the international community to pass through this difficulty.
Sample output 2	
Baseline	In an interview Hoffman CBS news magazine “60 minutes” ...
Bilingual LSA	Hoffman told the CBS news magazine “60 minutes” ...
Reference	Hoffman sighed when doing an interview with America's CBS news magazine “60 minutes”

Table 3.40: Examples demonstrating some degree of semantic paraphrasing with bilingual LSA.

The right hand of Table 3.39 shows the translation performance using the GALE Phase 2 SMT system. The performance trend is similar to the medium-scale system. Improvement in BLEU and NIST are observed after applying bilingual LSA-based language model adaptation and translation lexicon adaptation. The final gain is additive after applying all techniques together, yielding 2.5% relative improvement in BLEU compared to the unadapted baseline. The gain after adaptation is reduced and not

statistically significant at a 95% confidence interval [30.70%, 32.34%] confidence interval with respect to the unadapted baseline. On the other hand, the improvement in NIST follows a similar trend with 1.6% relative improvement compared to the unadapted baseline after applying language model adaptation, translation lexicon adaptation and tri-gram LSA. In addition, the gain is statistically significant at a 95% confidence interval [9.147, 9.378] with respect to the unadapted baseline.

3.4.3.4.5. End-to-End Translation Results

Table 3.41 shows the development of Mandarin-English speech translation using the GALE transcription and translation systems. Translation on the N -gram LSA-adapted word hypotheses with lower character error rate translates to better translation performance successfully, yielding 0.9% relative improvement in both BLEU and NIST compared to the background unadapted word hypotheses. Using manual transcription as inputs gives the best translation performance with 5.9% and 4.7% relative improvement in BLEU and NIST, respectively, compared to the background unadapted word hypotheses.

Although the Chinese side of the parallel corpora and the inputs were segmented using the Stanford segmenter consistently, the out-of-vocabulary (OOV) rate on the manual transcription was 1.7%, which was moderately high due to the ability of the Stanford segmenter to hypothesize new vocabulary during segmentation. Unfortunately, the out-of-vocabulary Chinese words cannot be translated, because of the possible segmentation mismatch with a phrase table. Therefore, we attempted to reduce the mismatch via segmentation refinement. First, we extracted the Chinese words from the phrase table to form a word list. Then, a maximal matching segmenter with this word list was applied on the Chinese test inputs that were segmented with a Stanford segmenter. As a result, the OOV terms were further segmented into smaller terms which may conform better to the segmentation of the phrase table. As shown in the second sub-table of Table 3.41, the OOV rate drops significantly to less than 0.1%. In addition, segmentation refinement translates to better translation performance, yielding 3% relative improvement in BLEU on the manual transcription compared to the corresponding counterpart with Stanford segmentation only. Similar results are obtained on the automatic transcription with 2.8% relative improvement in BLEU compared to the corresponding counterparts. In other words, the benefit of using N -gram LSA-adapted word hypotheses is maintained after segmentation refinement.

Source input	After Refining Chinese Word Segmentation		After LM Adaptation		After Lexical Adaptation		After LM + Lexical Adaptation			
	BLEU	NIST	BLEU	NIST	BLEU	NIST	BLEU	NIST		
ASR hypo (background)	15.92	5.28	16.36	5.28	16.67	5.34	16.52	5.32	16.88	5.37
ASR hypo (N-gram LSA)	16.07	5.33	16.52	5.34	16.80	5.37	16.90	5.40	17.18	5.43
Reference (OOV=1.7%)	16.86	5.53	17.37	5.53	17.74	5.59	17.72	5.58	17.99	5.63

Table 3.41: Speech translation results on the unsequestered broadcast news portion of the Mandarin Eval07 test set (Eval07u.BN) on BLEU and NIST using the GALE Phase 2 SMT System. Source inputs are word hypotheses from an unadapted background ASR (CER=5.6%), an N -gram LSA adapted (CER=5.2%), and a manual reference (CER=0%). Confusion-network-like English references are used for scoring.

After segmentation refinement, we applied cross-lingual adaptation on target language models, translation lexicons and phrase tables incrementally as shown in Table 3.41. Similar to the results on text translation, language model adaptation and translation model adaptation improve the translation performance individually. When both adapted models are applied simultaneously, the gain is additive, yielding 3.2 – 3.6% relative improvement in BLEU and 1.7 – 1.8% relative improvement in NIST compared to the corresponding counterparts of the unadapted SMT systems. However, the improvement is not statistically significant at a 95% confidence interval [15.12%, 17.65%] in BLEU and [5.100, 5.459] in NIST compared to the unadapted speech translation system. Overall, it is beneficial to apply topic adaptation to improve recognition accuracy that translates to better translation performance. In addition, the gain from better recognition accuracy and sharper translation models after topic adaptation are additive, yielding 1.8% and 1.1% relative improvement in BLEU and NIST, respectively, compared to the background unadapted word hypotheses. Comparing with the translation results on manual transcription, there is still much room for further improvement in speech translation. In other words, we expect that improving the upstream recognition accuracy will improve the downstream translation performance.

3.4.3.5 Conclusions

We have presented a unified unsupervised topic adaptation framework for monolingual and cross-lingual adaptation via latent semantic analysis. Our approach provides efficient topic inference which requires a small amount of text for rapid adaptation. Our contributions include the formulation of latent Dirichlet-Tree allocation that generalizes latent Dirichlet allocation for modeling topic correlation. In addition, we have extended LSA to N -gram LSA, derived a variational EM algorithm for efficient training, and formulated fractional Kneser-Ney smoothing to handle fractional counts. We have brought monolingual adaptation to cross-lingual adaptation via bilingual LSA. Thanks to a one-to-one topic correspondence, bilingual LSA facilitates cross-lingual topic adaptation using a source text to adapt the models on a target language before SMT decoding. In large-scale evaluation, we have shown significant performance improvement on recognition accuracy and translation performance, and an additive gain for end-to-end evaluation where improving the recognition accuracy of input word hypotheses has translated successfully to better translation performance in speech translation.

3.5 Language-Specific Models and Systems: Mandarin

3.5.1. Introduction

Author: Lidia Mangu

The goal of the Global Autonomous Language Exploitation Program is to empower the warfighter through the use of language technologies. In order to accomplish this goal,

the technologies must be integrated into operational engines that meet the needs of both warfighters and those who support them. That integration imperative leads directly to the two key parts of this section: how to perform that integration, and assessing how well we have met operational needs.

The speech-to-text (STT) task in the GALE Program commands highly sophisticated large vocabulary continuous speech recognition (LVCSR) systems that must integrate the latest technical advances in acoustic and language modeling, utilize multiple model sets and decoding setups, and optimize for the target languages and domains. Indeed, the development of the GALE Mandarin broadcast transcription systems covered in this section has led to the most authoritative Mandarin LVCSR work to date both in terms of system architecture and recognition performance.

There are four sections under this topic, discussing the respective work at IBM, Carnegie Mellon University, SRI/University of Washington, and Cambridge University.

The section by Chu *et al.* (IBM) presents an advanced multi-pass training and decoding framework, representative of the state-of-the-art, and demonstrates its application in the Rosetta Mandarin transcription system. As a distinctive discriminative feature in Chinese, tone receives a comprehensive treatment here with respect to feature extraction and normalization, information fusion, and phonetic modeling. In addition, they introduce a language model rescoring technique for topic adaptation, combining both static topic LMs trained a priori, and dynamic topic LMs trained on demand.

The section by Hsiao *et al.* (CMU) gives a detailed account of the speaker clustering and system combination techniques used in the Rosetta system. In particular, the six step lattice combination recipe is shown to give clear error reductions over the best single system hypotheses. Various improvements of the CMU Mandarin ASR system are also discussed here.

Lei *et al.* (SRI/UW) describe the Mandarin transcription system jointly developed by SRI and University of Washington. Moreover, they consider vocabulary expansion in Chinese and gives a simplified solution built upon existing work on English compound word extraction. To address word-level pronunciation variations, the authors propose a data-driven lexicon expansion method using forced alignment on character-level pronunciation networks. An effective LM adaptation method similar to that described in Chapter 3.1 is also implemented.

Tomalin *et al.* (CU) aim to quantify the relationship between the performance of speech recognition systems and that of a downstream machine translation (MT) system. The outputs from five incrementally improved STT systems are put through an MT system, and the impact of the incremental STT gains is measured in TER and BLEU scores. The authors show that there is an approximately linear relationship between STT and MT performance improvements. It is further shown that adding a post-STT sentence boundary detection stage can lead to additional gain in MT performance.

3.5.2. Towards the State of the Art in Automatic Mandarin Broadcast Speech Transcription

Authors: Stephen M. Chu, Hong-Kwang Kuo, Lidia Mangu, Qin Shi, Shilei Zhang, Yong Qin, Qin Jin, Ian Lane, and Yik-Cheung Tam

3.5.2.1 Introduction

This section describes the main development activities for Mandarin STT by the GALE Rosetta team. We introduce an advanced large vocabulary continuous speech recognition platform, and highlight technical advances in three areas that have led to notable improvement in recognition performance: tone modeling for Mandarin ASR; search and classification based language model adaptation; and lattice combination. The effectiveness of the proposed methods is demonstrated on our GALE Phase 2 evaluation system. We show that, combined, these advances lead to 34% and 16% relative reduction in character error rate for broadcast news and broadcast conversations, respectively. It is further shown that the resulting system achieves 7.2% and 6.4% CER on the two Phase 2 test sets, representing the state of the art in Mandarin broadcast speech transcription.

With the prevalence of discriminatively trained context dependent speaker adaptive acoustic models (Chen 2006), and the ever growing availability of training data, automatic speech recognition (ASR) systems have seen a significant boost in performance in recent years. In this section, we demonstrate the development of such a modern ASR system and its application in Mandarin broadcast speech transcription. Moreover, we shall focus on three additional techniques that are especially relevant to the GALE Mandarin STT task, and show that further reduction in error rate can be achieved over the state-of-the-art baseline.

First, the tone is a distinctive discriminative feature in Mandarin Chinese. We take the embedded approach to tone modeling and present an effective recipe for feature extraction/normalization, information fusion, and phonetic modeling. Second, compared to casual speech, broadcast material is more structured and much likelier to have well defined topics. This work demonstrates that this additional dimension can indeed be successfully exploited with topic-adaptive language model (LM) rescoring. Third, because low latency is generally not a critical factor in offline transcription tasks, multiple recognition systems can be combined both sequentially using cross-adaptive decoding, and in parallel using lattice combination to further reduce the recognition error rate.

3.5.2.2 System Architecture

The IBM baseline GALE Mandarin broadcast speech transcription system operates in multiple passes. Figure 3.20 shows the processing pipeline from the perspective of acoustic modeling.

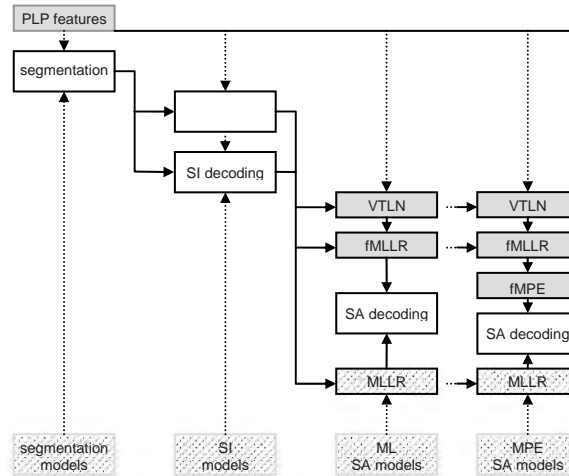


Figure 3.20: The IBM Mandarin broadcast speech transcription system consists of speech segmentation, speaker clustering, speaker independent (SI) decoding, and speaker adapted (SA) decoding.

3.5.2.2.1. Front End Processing

The basic features used for segmentation and recognition are perceptual linear prediction (PLP) features. Feature mean normalization is applied as follows: in segmentation and speaker clustering, the mean of the entire session is computed and subtracted; for SI decoding, speaker-level mean normalization is performed based on the speaker clustering output; and at SA stage, the features are mean and variance normalized for each speaker. Consecutive feature frames are spliced and then projected back to a lower dimensional space using linear discriminant analysis (LDA), which is followed by a maximum likelihood linear transform (MLLT) (Saon 2000) step to further condition the feature space for diagonal covariance Gaussian densities.

3.5.2.2.2. SI Models

Phones are represented as three-state, left-to-right HMMs. With the exception of silence and noise, the HMM states are context-dependent conditioned on quinphone context covering both past and future words. The context-dependent states are clustered into equivalence classes using a decision tree.

Emission distributions of the states are modeled using mixtures of diagonal covariance Gaussian densities. The allocation of mixture component to a given state is a function of the number of frames aligned to that state in the training data. Maximum likelihood (ML) training is initialized with state-level alignment of the training data given by an existing system. A mixture splitting algorithm iteratively grows the acoustic model from one component per state to its full size. One iteration of Viterbi training on word graphs is applied at the end.

3.5.2.2.3. SA Models

The SA acoustic models share the same basic topology with the SI model. For speaker adaptation, a model space method, *maximum likelihood linear regression* (MLLR), and two feature-space methods, *vocal tract length normalization* (VTLN) and *feature-space MLLR* (fMLLR) (Gales 1998), are used in the baseline system.

An eight level binary regression tree is used for MLLR, which is grown by successively splitting the nodes from the top using soft K-means algorithm. The VTLN frequency warping consists of a pool of 21 piecewise linear functions, or warping factors. In decoding, a warping factor is chosen such that it maximizes the likelihood of the observations given a voice model built on static features with full covariance Gaussian densities.

In addition to the speaker adaptation procedures, the improved Mandarin transcription system also employs the discriminately trained *minimum phone error* (MPE) (Povey 2002) models and the *feature-space MPE* (fMPE) (Povey 2005) transform. Here the training uses a new 2-stage process slightly different from the original recipe demonstrated by Povey. In the first stage, we estimate an fMPE transform from an existing set of models. In the second stage, we retrain the decision trees for context-dependent state clustering and estimate ML models in the new fMPE space. MPE training of the acoustic models is subsequently done in the new fMPE space.

We found that the 2-stage process consistently gives more than 0.5% absolute CER improvement in our Mandarin systems.

3.5.2.3 Tone Modeling

Treatments of tones in Mandarin ASR can be grouped into two categories: explicit tone recognition and embedded tone modeling. The latter approach, considered in this work, inherently allows joint optimization of tonal/atonal parameters in a coherent system by incorporating tonal components in both the acoustic models and the feature stream.

3.5.2.3.1. Tonal Phonetic Units

A straightforward tonal phone set can be derived directly from the *Pinyin* system, which maps Mandarin phonetics using a set of *initials* and *finals*. If we treat each initial and each tonal final as phonemes, the total number of phonemes is $N = C + T \times V$, where C is the number of consonant initials, T is the number of tones, and V is the number of vowel endings. This would lead to a large set of $20 + 5 \times 39 = 215$ phones.

By grouping the glides with the consonant initials into *premes*, each syllable can be decomposed into *demisyllables*. With the demisyllable system, N is reduced to: $53 + 5 \times 21 = 158$, where 53 is the number of premes, and 21 is the number of the second demisyllable without glide. Adding four auxiliary phonemes (undefined, word boundary, intra-word silence, and inter-utterance silence) brings the total number of phonemes in the set to 162. This tonal demisyllable system is used as our basic phonetic units for all systems described in this section.

Note that the number of phonemes can be drastically reduced if we assume that the pitch information on the main vowel is sufficient to determine the tone of a whole

syllable. We compared a system built on the resulting 77-phone set and the demisyllable system, and found the latter to be superior with large training sets, thus more suitable to the GALE scenario.

Furthermore, the context-dependent phones in our tonal system use decision trees grown with tone-specific questions. This in effect allows syllables that differ only in tones to share parameters, and gives a data-driven way to assign the importance of tonal identity of a particular context-dependent acoustic unit.

3.5.2.3.2. Tonal Features

The frontend described in Section 3.5.2.2.1, which is used in our baseline system, does not capture pitch information in speech. The tonal system augments the standard frontend with explicit pitch features extracted using the YIN (de Cheveigné 2002) algorithm.

The pitch tracker gives two outputs: an estimated pitch contour of the audio signal, and a frame-level confidence score given by the algorithm, as shown in Figure 3.21 (b). The pitch is undefined in unvoiced region and during silences, and can thus cause large noise spikes in the estimation, which, if left untreated, will have an adverse effect on statistical modeling. To address this problem, we first use the confidence score to classify voiced and unvoiced regions of the speech signal. Then the pitch values in the unvoiced segments are replaced through *piecewise cubic Hermite interpolating polynomial* using samples in the adjacent voiced regions. The procedure and the result are demonstrated in Figure 3.21(a). Further smoothing and normalization are applied using median and moving average filtering.

Because the confidence score provides strong cues for voicing, we also add it as an additional feature dimension; together with the smoothed pitch contour, a two dimensional tonal feature vector is thus constructed.

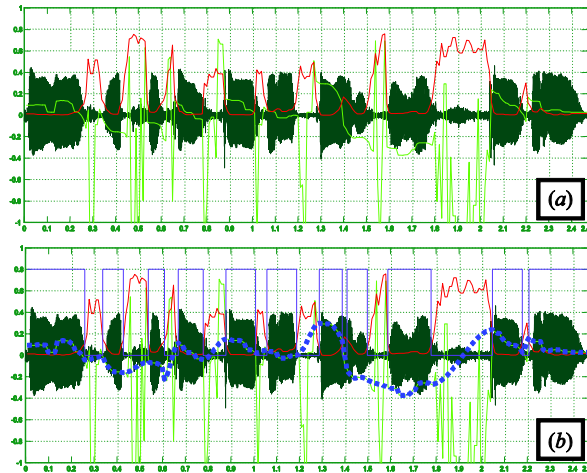


Figure 3.21: (a) Pitch tracker output, where the green/light-gray line is the estimated pitch value and the red/dark-gray line is the confidence score (lower is more confident). (b) Interpolation and smoothing results. The square signal shows the voiced/unvoiced classification, and the dotted line is the pitch feature after interpolation.

3.5.2.3.3. Feature Fusion

The five tones in Mandarin are defined by relative pitch movement rather than absolute frequency value. Therefore, the raw pitch features should be processed such that the underlying dynamics can be captured.

One way is to fuse the tonal features with the standard PLP features *early*, so that the subsequent splicing and LDA transforms can extract the dynamics in a data-driven manner. The other is to compute the Δ and $\Delta\Delta$ of the pitch first and integrate them with the regular feature stream *late* after the LDA projection. We implemented both methods and found that the *early* fusion approach gained more from the tonal features. Results on the tonal system reported in this section are from the first implementation.

3.5.2.4 Topic-Adaptive LM Rescoring

Adaptation techniques in language modeling have shown growing potentials in improving speech recognition performance. For topic adaptation, a set of pre-defined topic-specific language models are typically used, and adaptation is achieved through adjusting the interpolation weights. However, mismatch between the test data and the pre-defined models inevitably exists and is left untreated in the static approach. Instead of tuning the parameters in the existing models, this section presents a method that dynamically extracts relevant documents from training sources according to intermediate decoding hypotheses to build new targeted language models. Different from general search-based document collection, a new and effective ranking method is used here for candidate extraction. The targeted language models are interpolated with the static topic language models and a general language model, and used for lattice rescoring.

3.5.2.4.1. Corpus Classification Using SVM

Topic based language modeling relies on the assumption that the usage of content words typically depends on the topic of the text. A key step in constructing topic-specific LMs is to cluster the corpus into the appropriate topics.

The SVM, which has been shown to yield good generalization performance on a wide variety of classification problems with large-scale input space, is employed in our system to carry out text classification and topic clustering.

Manually labeled documents are used to train the SVM. The feature vector representing each training sample is a vector of terms given by our Mandarin text segmenter. To reduce the number of nuisance features, words occurring in fewer than three documents are omitted, and stop words are deleted from the feature vector. The overall classification accuracy of the topic classifier as measured by the F_1 measure is 0.8. Based on the SVM classifier, the entire LM training data was partitioned into 129 topics, and 129 *static topic LMs* are built based on the clustered training documents.

3.5.2.4.2. Search-Based LM

Although static topic language models can help to improve recognition accuracy through modeling different distributions of content words among topics, such topic models might still be too general for a specific story. For instance, under the broad topic class of politics, there are many sub-topics that are not related to the current story. In order to make sure that the n -grams related to the story are promoted, the ideal scenario is that we can find documents whose content is closely matched to the specific story.

The ranking algorithm used in the search engine is a combination of *vector space model* (VSM), a common information retrieval method, and the Boolean model to determine how relevant a given document is to a user's query. The motivation of VSM is that the more times a query term appears in a document relative to the number of times the term appears in all the documents in the collection, the more relevant that document is to the query. The Boolean model is used to first narrow down the documents that need to be scored based on the use of Boolean logic in the query specification.

Note that search here is different from a general information retrieval case in two ways. The first difference is that there are errors in the ASR transcription used to generate query terms. Such errors will misguide the search engine. The second difference is that the search should adhere to similarity at the document level. Thus, special attention must be paid to search key selection, and the scoring method should be modified accordingly.

A topic T_k is composed of m utterances: (u_1, u_2, \dots, u_m) , where an utterance u_i is expressed as: $u_i = (w_{1i}, w_{2i}, \dots, w_{ni})$, and w_{ij} represents a word. Silence and hesitation words in the utterance are tagged as potential chop points, thus u_j can be decomposed into word segments at these boundaries. After discarding very short segments, the word list is converted into a search key list $(key_{1j}, key_{2j}, \dots, key_{lj})$. For each search key key_{ij} , a large number of documents are typically retrieved, which are first sorted by the *ranking score*, and M documents with the highest scores are kept. After collecting all the documents selected by the keys in one topic, we define another score called the *hit-rate score* for each document, which is defined as the number of keys in the topic that occur in the document. According to this score, the documents are sorted again. Intuitively, the more similar the document is to the topic, the higher the document's *hit-rate score* would be. The *hit-rate score* also helps to reduce/prevent irrelevant documents from being extracted due to ASR errors in the search keys. Finally, a given number of documents with the highest hit-rate scores are collected to build a 4-gram LM. We shall refer this LM as the *search-based topic LM*. Figure 3.22 shows the re-ranking method.

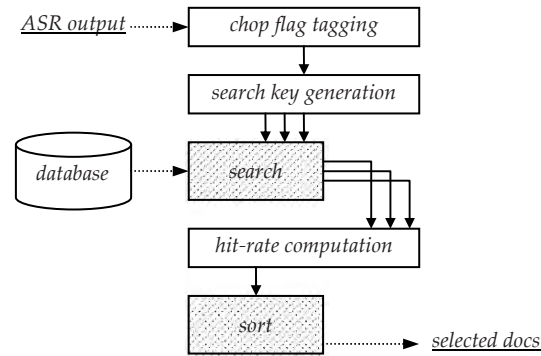


Figure 3.22: The documents selection algorithm uses the ranking score and the hit-rate score to select and prune the relevant documents.

3.5.2.4.3. Interpolation Method

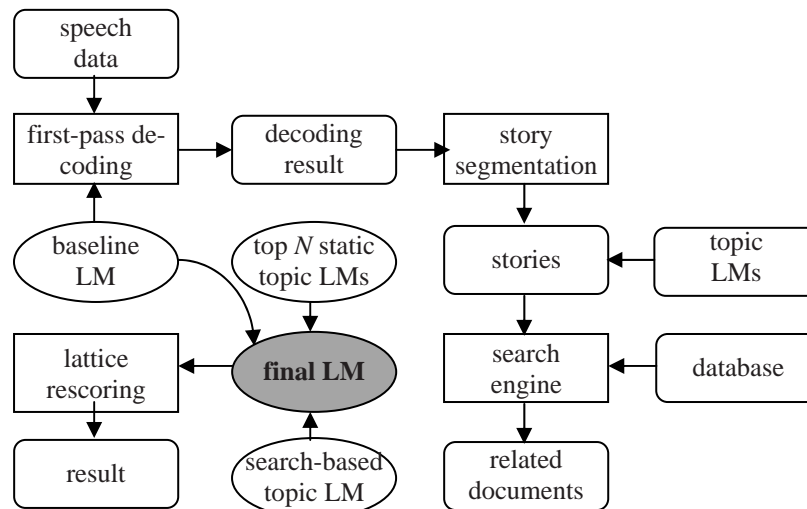


Figure 3.23: The proposed LM adaptation algorithm interpolates static topic-LMs, the search-based topic LM, and the general baseline LM for lattice rescoring.

When decoding a segment of speech that is assumed to be of a single topic, three kinds of language models can be combined: (1) the baseline language model, (2) static topic language model, and (3) search-based topic language model. To efficiently interpolate all the models, not all static topic language models are involved. Using the perplexity of the first pass decoding result with respect to different topic LMs, a subset of the static topic LMs with the lowest perplexity scores are selected as the candidates to interpolate with other models. The interpolation weights are chosen to optimize perplexity based on the first pass decoding result. Figure 3.23 shows the entire topic adaptation process.

3.5.2.5 Lattice Combination

To utilize multiple STT systems, we applied an effective lattice combination technique developed at the Carnegie Mellon University. Details of the method can be found in Section 3.5.3 (Hsiao *et al.*).

3.5.2.6 Experiments

The techniques described above were integrated in the Rosetta Mandarin transcription system for the GALE Phase 2 evaluation.

3.5.2.6.1. System Setup

For the baseline system, the 16 KHz input signal is coded using 13-dimensional PLP features with a 25ms window and 10ms frame-shift; 9 consecutive frames are spliced and projected to 40 dimensions using LDA. For the tonal system, the 2-dimensional tonal features are appended to the PLP features to form a 15-dimensional feature vector, and the rest of the processing pipeline is unchanged.

All acoustic model trainings use the same set of 1,737 hours of acoustic training data released by LDC for the GALE program. The SI model has 10K quinphone states modeled by 300K Gaussian densities. The SA model uses a larger tree with 15K states and 800K Gaussians.

The decoding LM is built by interpolating 20 backoff 4-gram models using modified Kneser-Ney smoothing. The interpolation weights are chosen to optimize the perplexity of a 364K held-out set. In total, 5GB of text data is used in training. The final LM has 6.1M *n*-grams and 107K words.

The topic-adapted LM for lattice rescoring is obtained by interpolating the *static topic LMs*, the *search-based topic LM*, and the general LM.

For the *static topic LMs*, the topics are organized as a manually constructed tree with 129 leaf nodes. More than 20,000 Chinese news articles related to the topics are collected and annotated. Using the SVM classifier, the entire LM training data was partitioned into 129 topics. And, 129 language models are built based on the documents belong to the topics.

For the *search-based topic LM*, an offline database with data collected before the GALE specified cut off date (Oct. 31, 2006) is constructed for search in decoding time.

3.5.2.6.2. Phase 2 Results

The test set for GALE 2 evaluation contains Mandarin broadcast audio collected from TV programs in mainland China, Taiwan, and Hong Kong. It consists of two subsets. The first is the sequestered part of the Phase 2 evaluation set, referred to as *eval'07* here. It contains 1 hour and 11 minutes of audio, and has 20.6K characters in the reference transcript. The second set, denoted *brdg'07*, covers the additional data defined for Phase 2. This set contains 1 hour and 13 minutes of audio and 21.0K characters in the reference. We further divide the experimental results on *eval'07* into broadcast news (*bn*) and broadcast conversations (*bc*) for more in-depth analyses.

The results are summarized in Table 3.42. First, the tonal system shows clear improvement over the baseline on both sets. Especially for *bn*, the tonal system manages to reduce the CER from an already low baseline of 3.8% down to 3.3%. Clearly, *bn* content benefits more from tone modeling than *bc*. One possible reason is that the style of speech in *bc* makes it harder to extract tonal information reliably. Moreover, the *bc* portion of the data is much more likely to contain accented speakers, and incorrect tone pronunciation is often the first degradation of non-standard Mandarin.

	<i>brdg'07</i>	<i>eval'07</i>	<i>bn</i> ,	<i>bc</i>
BASE:	9.0	8.1	3.8,	13.3
TONE:	8.8	7.8	3.3,	13.2
BASE \times TONE:	8.4	7.5	3.3,	12.6
X_a :	8.4	7.4	3.2,	12.5
X_b :	8.2	7.4	3.4,	12.3
Topic LM + X_a :	7.4	6.5	2.5,	11.3
Topic LM + X_b :	7.4	6.7	2.6,	11.6
LC:	7.2	6.4	2.5,	11.2

Table 3.42: Summary of the phase 2 evaluation results.

Adapting the tonal system from the baseline output leads to further improvement. Overall, introducing tone modeling into the pipeline gives 0.6% absolute error reduction on both sets.

Cross-adaptive decoding is subsequently carried out with the two IBM systems and a Mandarin STT system developed at CMU. The description of the CMU system is covered by Hsiao *et al.* in Section 3.5.3. The CERs of the CMU system on *brdg'07* and *eval'07* are 11.6% and 11.5%, respectively. Two set's of lattices, X_a and X_b are produced by the three-way cross-adaptations, as shown in Figure 3.24. The particular combinations are chosen such that: 1. a better CER on the development set is achieved; and, 2. the resulting lattices are sufficiently different.

LM rescoreing and topic adaptation on the two sets of cross-adapted lattices give 0.7 to 1.0 improvements. To understand how much of the gain can be attributed to topic adaptation, a quick comparison is made with non-adaptive lattice rescoreing using a large LM. On *eval'07*, this non-adaptive method gives 0.3% CER reduction. This suggests that at majority of the gain in this step comes from the topic adaptation process.

Finally, lattice combination reduces the CER further down to 7.2% and 6.4% for the two test sets. Compared with the baseline, 20% and 21% relative error reductions have been achieved.

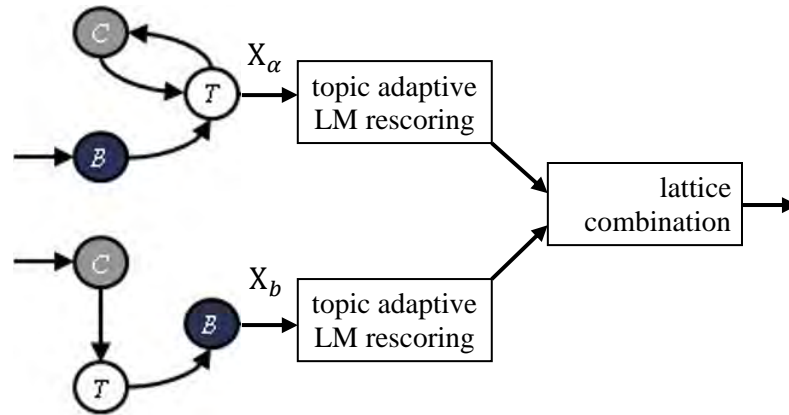


Figure 3.24: STT pipeline used in the phase 2 Mandarin evaluation. *B*: IBM baseline system, *C*: CMU system, and *T*: IBM tonal system.

3.5.2.7 Conclusions

This section considers the Mandarin broadcast speech transcription task in the context of the DARPA GALE project. We present a state-of-the-art Mandarin STT architecture and describe the technical improvements made through the Phase 2 evaluation.

3.5.3. The CMU-InterACT Mandarin Transcription System for GALE

Authors: Roger Hsiao, Mark Fuhs, Yik-Cheung Tam, Qin Jin, Ian Lane, and Tanja Schultz

3.5.3.1 Introduction

We present the 2007 CMU-InterACT Mandarin speech-to-text transcription system. Our system employs a 3-pass decoding strategy, incorporating discriminatively trained acoustic models trained with over 1300 hours of transcribed audio. To better handle the broad domain of topics that occur in broadcast speech, a novel unsupervised language model adaptation scheme based on latent semantic analysis was employed. We have also developed a novel lattice-combination approach, which was designed specifically for spoken language translation. In the GALE Phase 2 evaluation, our system achieved a character error rate (CER) of 12.9% on dev07 and 13.3% on the eval07 unsequenced test set.

This section describes our efforts developing the CMU-InterACT Mandarin speech-to-text (STT) transcription system within the GALE Program. During the first two years of the program we explored four specific research areas to improve transcription accuracy, specifically; audio segmentation and speaker clustering, discriminative training of acoustic models, unsupervised language model adaptation, and STT system combination techniques.

First, we extended our earlier works in audio segmentation and speaker clustering (Jin and Schultz 2004) and applied these to the GALE task. Using these approaches

transcription accuracy degraded only slightly compared to using manually segmented input, showing the effectiveness of these techniques. Second, we incorporated discriminative training during acoustic model building. We investigated two algorithms; maximum mutual information estimation (MMIE) and boosted maximum mutual information estimation (BMMIE). Both techniques were found to be effective, and discriminative training obtained the largest improvement of any single technique. Our final acoustic models were successfully trained on over 1300 hours of transcribed speech, showing the effectiveness of these approaches on large scale tasks. Third, to better handle the broad domain of topics that occur in broadcast media, we developed a novel unsupervised language model adaptation scheme based on latent semantic analysis (LSA) (Tam and Schultz 2005; Tam and Schultz 2006). This approach further improved transcription accuracy and was successfully applied in both lattice rescoring and automatic speech recognition (ASR) decoding. Finally, to better leverage multiple STT engines, we developed a novel lattice combination approach which improved transcription accuracy while retaining time-coherent hypotheses, which are better suited for spoken language translation.

3.5.3.2 Audio Segmentation and Clustering

In our speaker segmentation and clustering system, audio segmentation is realized by an HMM segmenter with four classes: Speech, Noise, Silence, and Music. The Speech features used are 13-dimension MFCCs plus their first and second derivatives. Each class is represented by a GMM with 64 Gaussians which were trained on 3 hours of manually annotated HUB4 shows. The resulting speech segments are then automatically clustered, ideally generalizing a style cluster corresponding to each individual speaker. During clustering, a hierarchical, agglomerative clustering technique with Bayesian Information Criterion (BIC) stopping criteria is used (Jin and Schultz 2004). A tied Gaussian mixture model (TGMM) is built on the whole set of speech segments. A GMM for each cluster is then trained by adapting the TGMM. Initially, each segment is considered as a separate cluster. We define the distance between two clusters by the Generalized Likelihood Ratio (GLR):

$$D(C_1, C_2) = -\log \frac{P(X|\theta)}{P(X_1|\theta_1)P(X_2|\theta_2)} \quad (3.42)$$

where X_1 , X_2 , and X are feature vectors in cluster C_1 , in cluster C_2 , and in the merged cluster of C_1 and C_2 , respectively. θ_1 , θ_2 , and θ are statistical models built on X_1 , X_2 , and X , respectively. At each iteration of agglomerative clustering, the two closest clusters are merged and a model of the new cluster is re-estimated. Clustering is repeated until the BIC stopping threshold is exceeded.

Model	dev07	eval07unseq
BN-manual	7.3	5.3
BN-auto	7.7	6.0
BC-manual	18.0	24.1
BC-auto	19.5	25.7
All-manual	13.3	13.5
All-auto	14.4	14.5

Table 3.43: Comparison on using manual and automatic segmentation in CER.

In the 2006 GALE evaluation, the test data consisted of one snippet per show with a duration of 3 - 5 minutes. However, in the 2007 evaluation, snippets were 1 - 2 minutes long, and shows contained multiple snippets. Since it is reasonable to assume that the same speakers may occur in different snippets within the same show, we conducted clustering at the show level rather than at the snippet level. To ensure that we do not underestimate the number of speakers in multiple-snippet shows, two different BIC thresholds were used dependent on the number of snippets per show. A comparison of our automatic segmentation to the reference segmentation is given in Table 3.43.

3.5.3.3 Acoustic and Language Modeling

Feature extraction employs standard 13-dimension MFCC features extracted using a 16ms window with 10ms frame shift. 15 adjacent feature vectors are concatenated and linear discriminant analysis is then applied to reduce the feature dimension to 42. We apply cepstral mean and variance normalization as well as vocal-tract length normalization per speaker.

In our legacy RT04 system (Yu *et al.* 2004), we found that cross adapting an Initial-Final (I-F) acoustic model on the output of a phone-based decoder improved recognition accuracy compared to using multiple systems with the same phonetic units. We apply a similar architecture for our GALE system.

The CMU Mandarin transcription system is a 3-pass system. It consists of three sets of acoustic models. The first acoustic model (AM1) is a speaker independent (SI) I-F model using multi-style training by uniformly mixing the BN and BC shows together. The second acoustic model (AM2) is a phone model using speaker-adaptive training (SAT) with feature space adaptation (FSA). AM2 uses hypotheses from AM1 for cross adaptation. Instead of using multi-style training as with AM1, AM2 is genre dependent: its senone tree considers whether the incoming show is BN or BC which is determined on the show name. The third acoustic model (AM3) is also speaker adaptive, based on the output of AM2, but AM3 uses I-F models. AM3 is trained using maximum mutual information estimation (Valtchev *et al.* 1997) (MMIE) and boosted maximum mutual information estimation (BMMIE). Table 3.44 gives a summary of our acoustic model settings.

Model	AM1	AM2	AM3
Unit	I-F	phone	I-F
model type	SI	SAT-FSA	SAT-FSA
training	ML	ML	(B)MMIE
# codebooks	6K	10K	10K
genre-dep	no	yes	no
algorithms	-	STC/VTLN	STC/VTLN

Table 3.44: Acoustic Model Configurations

During acoustic model training, we cluster states using a quinphone decision tree with tonal questions incorporated such that a single tree implicitly models all tonal variants of the same base phone/syllable. The number of Gaussian mixtures per state is determined using merge and split (MAS) training with a maximum of 100 mixtures per state. We then apply a global semi-tied covariance (STC) matrix (Gales 1999) on all the acoustic models.

The acoustic model training corpora consist of over 1300 hours of transcribed audio data released within the GALE program containing both BN and BC shows. The BN sources are mainly from CCTV, NTDTV, PhoenixTV and VOA, while the BC sources are mainly from CCTV and PhoenixTV.

3.5.3.3.1. Discriminative Training on Acoustic Models

We apply MMIE (Valtchev *et al.* 1997) and BM-MIE (Povey *et al.* 2008) on AM3 in order to improve recognition accuracy. BMMIE is an extension to MMIE proposed by Povey *et al.* (2008), which considers the raw phone accuracies of the competing word arcs in the lattices. Detailed description of BMMIE and comparison between BMMIE and MMIE is available in Section 3.3.4 by Povey *et al.* In our system, the boosting factor of BMMIE is 0.5 and we apply I-smoothing (Povey 2003) with $\tau = 100$ for both MMIE and BMMIE training. ML estimate is used for I-smoothing.

3.5.3.3.2. Language Modeling

The LM training corpora consists of over one billion word tokens obtained from the Mandarin Gigaword V2, audio training transcripts and the web data released within the GALE Program. Documents are first divided according to their news sources and a 4-gram LM is built for each source using the modified Kneser-Ney smoothing scheme using the SRI LM toolkit (Stolcke 2002). A background LM is generated by linearly interpolating the source-dependent language models using the dev07 development set. Dynamic LM adaptation is applied using correlated latent semantic analysis (LSA) (Tam and Schultz 2007) to model topic correlation. In LSA training, a set of topic-dependent unigram LMs are built. In testing, the topic weights θ_k are incrementally adapted using the previously decoded utterances within the same show via a variational E-step $P_{lsa}(w) = \sum_{k=1}^K \hat{\theta}_k \cdot P(w|k)$ with $\hat{\theta}_k = \frac{\gamma_k}{\sum_{k=1}^K \gamma_k}$ where γ_k denotes the fractional posterior count of topic k inferred from the word context. The adaptive LSA unigram is then log-linearly interpolated with the background LM to decode the next utterance. The adaptive LSA unigram works like a cache-based LM. However, instead of caching the word counts in the word history, latent topic counts are cached. Thus, this approach is shown to be robust against speech recognition errors.

Marginal LM adaptation (Kneser *et al.* 1997) is an effective technique grounded by information theory where the background LM is adapted so that its unigram marginals match the in-domain unigram marginals. In our work, the in-domain marginals are

estimated by $P_{lsa}(w)$. The adapted LM has the form: $Pr_a(w|h) = \frac{\alpha(w) \cdot Pr_{bg}(w|h)}{Z(h)}$ where $Z(h)$ is a normalization term to guarantee that the probability sums to unity. $\alpha(w)$ is a scaling factor, which is commonly approximated as: $\alpha(w) \approx \left(\frac{Pr_{lsa}(w)}{Pr_{bg}(w)}\right)^\beta$. This technique is expensive for decoding due to computing the normalization term $Z(h)$. On the other hand, this approach can be easily applied during lattice rescoring since only the outgoing word links of the same context node are considered when computing $Z(h)$. The LM score of a word link (i, j) can be adapted analogously as follows: $lm_a(i, j) = Mass(i) \cdot \frac{\alpha(w(i, j)) \cdot lm_{bg}(i, j)}{\sum_{o \in Out(i)} \alpha(w(i, o)) \cdot lm_{bg}(i, o)}$ where (i, j) denotes a link from node i to node j and $w(i, j)$ is the word label associated to this link. $Mass(i)$ denotes the total probability mass of the outgoing links in $Out(i)$ coming from the same context node i . Using our GALE-2006 Mandarin evaluation system, 0.7% absolute reduction in character error rate (CER) is observed on the eval06 development set by incorporating LSA-based LM adaptation compared to the unadapted LM baseline which obtained CER of 20.4%.

A recent extension of this approach is bigram LSA which relaxes the “bag-of-word” assumption present in LSA. In addition to LSA, bigram LSA is applied during for lattice rescoring.

3.5.3.4 Evaluation Setup

In the GALE 2 evaluation, dev07 and the unsequestered portion of eval07 were used as internal evaluation sets. Dev07 consists of 2.5 hours of audio from various channels, including CCTV, NT-DTV, and PhoenixTV. Eval07 unsequestered portion has 1.2 hours of audio from CCTV, NTDTV, PhoenixTV and AnhuiTV. Both dev07 and eval07 unsequestered portion consists of roughly 50% BN and 50% BC shows. Manual segmentation was provided, but speaker IDs had to be automatically generated. We performed speaker clustering using the procedure described in Section 3.5.3.2. Compared with IBM’s speaker clustering procedure, we found our approach lead to better recognition accuracy when ASR was performed with a preliminary system developed at IBM. Compared to the IBM system, CER was reduced by 0.2% and 0.6% on dev07 and eval07, respectively. The results reported in this chapter use CMU clustering.

Table 3.43 shows the performance of our system using manual and automatic segmentations without bi-gram LSA lattice rescoring for both segmentations, we performed speaker clustering using the approach discussed in Section 3.5.3.2. The manual segmentation is around 1% absolute better than automatic segmentation, which suggests the automatic segmentation of our system performs quite well. If we look at the performance break down, we can see that the gap on the BC shows is significantly larger. This is expected, since BC shows contain more complicated speaker turns and less fluent speech.

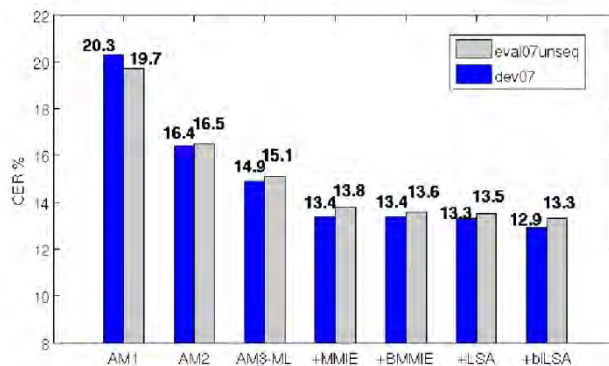


Figure 3.25: CMU Mandarin system performance at different development stages on the dev07 and eval07 (unsequestered portion) test set.

Figure 3.25 shows the performance at different stages of our system. Our system benefits from cross adapting I-F and genre dependent phone systems. After we obtained the ML model, we performed MMIE until the improvement converges on dev07. Due to time constraints leading up to the evaluation, we were only able to run one iteration of BMMIE on top of the MMIE model. However, BMMIE still improved the system slightly. With the addition of LM adaptation using LSA and lattice rescoring using bi-gram LSA adaptation, our system has a CER of 12.9% on dev07 and 13.3% on the eval07 unsequestered portion. We expect further improvement by applying BMMIE training directly to the ML model.

3.5.3.5 System Combination

Combining multiple systems together via cross-adaptation or confusion network combination (CNC) are established methods to improve recognition accuracy. We have found that the recognition accuracy of a group of STT systems can be improved by applying both techniques serially.

Confusion network combination (CNC) (Mangu *et al.* 2000) is an established method of improving automatic speech recognition accuracy. However, this and related approaches discard temporal constraints in the original input. The generated hypotheses can, therefore, violate the word order of the original inputs, which may degrade downstream natural language processing components. To overcome this limitation, we have developed a novel method to directly combine multiple lattices, while retaining temporal word order. Thus, the final hypothesis is time-coherent, and is more suited to post-STT sentence segmentation, punctuation restoration, and machine translation.

For a set of speech recognition word-lattices where links denote words and nodes indicate word boundaries with time markers, lattice-combination (LC) proceeds in six steps:

1. Calculate posterior probabilities of links in each lattice for use as confidence scores.
2. Scale confidence scores based on assigned system weights.
3. Group lattice links from all lattices into equivalence classes (as if for a CN).
4. Agglomeratively cluster lattice nodes in all lattices together by their similarity in time. Merge each cluster into a single node.
5. For links in each equivalence class, combine scores of multiple links with the same word.
6. Find the path that maximizes the minimum confidence over all sub-paths.

Significant improvements were observed only when the lattice nodes were aggressively merged, permitting paths through the combined lattice that were composed of links from multiple source lattices. Since the starting and ending times of the same (or very similar) words in different lattices invariably differed, nodes with similar times were agglomeratively clustered together until any further cluster joins would result in a cluster whose temporal span (node with highest time minus node with lowest time) exceeded a specified maximum. Increasing the maximum temporal span increases merging of the lattices, which permits a larger hypothesis space to be considered, however, it also decreases the temporal precision of the resulting hypothesis. For the results presented here, a maximum temporal span of 100ms was used, so word boundaries of the resulting hypothesis were guaranteed to be within 50ms of their original time in the lattice.

Very short links (short words, silence tokens, partial words, etc.), i.e., those that occupy less time than the clustering procedure's maximum temporal span, can cause a variety of abnormalities in the merged lattice. Most obviously, some form of loop detection is required to assure that a very short links start and end nodes are not merged together. More generally, the global ordering of links provided by the equivalence classes may be violated by the clustering procedure. This often occurs when one node with an incoming link in equivalence class i is merged with another node with an outgoing link in equivalence class i . To prevent these problems, each cluster is assigned values indicating the Latest Equivalence Class Before (LECB) among all incoming links and the Earliest Equivalence Class After (EECA) among all outgoing links. In order for two clusters to be merged, the following conditions must be satisfied: Cluster 1 LECB < Cluster 2 EECA and Cluster 2 LECB < Cluster 1 EECA. When two clusters are merged, the higher LECB and lower EECA are adopted for the new cluster.

System	CNC		LC	
	dev07	eval07	dev07	eval07
CxTxB	10.32	10.66	10.31	10.68
BxTxC	9.44	9.74	9.50	9.76
TxBxC	9.34	9.51	9.34	9.41
TxBxC BxTxC	9.03	9.28	9.03	9.29
TxBxC CxTxB	9.07	9.18	8.94	9.17
TxBxC CxTxB BxTxC	8.92	9.14	8.86	9.03

Table 3.45: Results (CER) of GALE phase 2 comparing various combinations of three Mandarin systems: C = CMU, B = IBM non-tonal (base), T = IBM tonal. CxTxB indicates the CMU system cross-adapted on output of the IBM tonal system, which was cross-adapted on the output of the IBM non-tonal system; others are similar.

In a traditional word lattice, the log acoustic and LM scores of different links are treated as (conditionally) independent and summed. In a merged lattice, a link is scored with a sum of posterior probabilities, which has a strong mutual dependence with other link scores before and after. In a CN, this mutual dependence is leveraged to maximize the confidence within each equivalence class independently, sacrificing any temporal constraints between classes. To overcome this mutual dependence in a merged lattice, the path was selected that maximized the minimum confidence score along all sub-paths. A divide-and-conquer strategy was used:

1. Given a start node i and end node j somewhere in the lattice, find the link (with start node m and end node n) limiting the single-path max flow of confidence from node i to node j .
2. Recursively call this procedure to find the best path from i to m and n to j .
3. Return a path consisting of the best path from i to m , the limiting link, and the best path from n to j .

This procedure was initiated with the initial and final nodes of the lattice. Table 3.45 compares the results of CNC and LC on three systems. Two of the systems were built by IBM, and all three systems were cross-adapted against one another prior to combination. The performance of CNC and LC is very similar, and this parity was observed on Arabic language systems as well.

In the GALE Phase 3 evaluation, ten STT engines were developed within the Rosetta team. Individual systems differed by the phonetic unit used in acoustic modeling, the explicit modeling of tones, cross-adaptation between systems, and the method of language model rescoring applied. LM rescoring techniques investigated included: Bi-gram LSA based LM rescoring (Tam *et al.*), Neural Network (NN) based LM rescoring (Emami *et al.*), and Topic-adaptive LM rescoring (Chu *et al.*). These techniques are described elsewhere in this chapter.

On the *dev08* development set these individual systems obtained CERs ranging from 6.7% to 8.7%. A list of the top performing systems is shown in

Table 3.46. In an attempt to improve the transcription accuracy above that obtained by the single best system “*Xa with NN and bi-gram LSA based LM rescoring*”, we applied the lattice combination approach described above.

System combinations	Character-Error-Rate % (Sentence Error Rate %)		
	Tuning	unseen test	
	dev-08	eval-07	retest-07
1	6.7 (59.9)	8.1 (57.6)	6.5 (51.5)
2	6.9 (60.3)	8.4 (59.0)	6.6 (52.3)
3	6.9 (59.8)	8.4 (59.0)	6.6 (52.3)
4	7.0 (59.9)	8.4 (59.4)	6.7 (54.1)
2 4	6.5 (59.4)	8.1 (56.3)	6.3 (52.6)
2 4 1	6.5 (59.9)	8.0 (57.1)	6.4 (51.6)
2 4 1 3	6.5 (59.1)	8.2 (57.3)	6.4 (52.0)

Table 3.46: Results of top performing ROSETTA phase-3 systems and lattice combination of these systems: 1 = X_a with NN and bi-gram LSA based LM rescoring, 2 = X_a with Topic-adaptive and NN LM rescoring, 3 = X_b with Topic-adaptive and NN LM rescoring, 4 = X_b with Topic-adaptive LM rescoring. X_a and X_b , are the system outputs defined in Section 3.5.2.6.2 (Chu *et al.*).

Due to the large number of STT engines, evaluating all possible n -way combinations is computationally expensive, thus, we applied an iterative selection approach. First, all pair-wise system combinations are evaluated, and the pair with minimum CER is selected as the initial combination-set, in this step $n = 2$. Next, the remaining systems are combined with the current best n -way combination and the $n + 1$ combination with minimum CER is selected. This process is iteratively repeated until the addition of more systems does not improve performance. During the selection process, system weights were optimized on a tuning set (*dev08*), and the combination that obtained the best performance on unseen data (*eval07* and *retest07*) was selected.

Using the proposed selection scheme, the best 2, 3 and 4-way system combinations identified in

Table 3.46 were selected. Although the 4-way combination “2 4 1 3”, obtained a slightly lower SER on the tuning set, higher error-rates were obtained on unseen data. Thus, iterative selection stopped at the 3-way combination “2 4 1”. Compared to the single best system, lattice combination reduced CER on all test-sets between 0.1 to 0.2 points absolute, a 3% relative reduction in CER.

3.5.3.6 Conclusion and Future Work

We present recent developments on the CMU Mandarin BN/BC transcription system. Our system achieved good performance on a large-scale evaluation by using MMIE- and BMMIE-based discriminative training, unsupervised LM adaptation using LSA and lattice-based system combination. Our future development includes feature space discriminative training, improved LSA modeling, better system combination and tight coupling between the ASR and machine translation systems.

3.5.4. Advances in Mandarin Speech-to-Text Transcription for Broadcast News and Conversations

Authors: Xin Lei, Wen Wang, Arindam Mandal, Andreas Stolcke, Mei-Yuh Hwang, Wei Wu, and Mari Ostendorf

3.5.4.1 Introduction

This section describes the recent advances in SRI-UW Mandarin broadcast news (BN) and conversations (BC) speech-to-text (STT) system, developed under the DARPA Global Autonomous Language Exploitation (GALE) Program. It is demonstrated that the core technologies for English STT such as discriminative training (Povey and Woodland 2002) and acoustic adaptation (Gales 1998) are also applicable to the Mandarin language. Mandarin specific components can be used to exploit the characteristics of the Chinese language. In our previous work, we have investigated pronunciation lexicon designs and lexical word segmentation in Hwang *et al.* (2009), as well as Mandarin tone modeling in Lei *et al.* (2006) to further improve the system performance. In this study, we give an overview of our Mandarin STT system and describe recent progress for GALE 2008 Evaluation.

3.5.4.2 Speech and Text Corpora

For acoustic model training, we use about 1642 hours of BN and BC speech data collected by LDC, including Mandarin Hub4 (30 hours), GALE Year 1 (747 hours), GALE Year 2 (394 hours), GALE Year 3 (317 hours) and GALE Year 4 Release 1 (154 hours) corpora. The amount of BN data is close to that of BC data. Training data are collected from broadcast stations such as CCTV, RFA, NT-DTV and PHOENIX, spanning 1997 through April 2008. Comparing to the December 2007 evaluation, we have removed 89 hours of TDT4 data that are relatively noisy and hurt performance slightly, and added 230 hours of Year 3 and Year 4 data.

We evaluate our system on two development test sets: DARPA GALE 2007 development set (dev07) and 2008 development set (dev08). The dev07 test set contains 2.4 hours of speech data from 74 broadcast shows, with 1.0 hour in BN and 1.4 hours in BC. The dev08 test set has 1.0 hour of speech from 44 shows, with 0.5 hour in both BN and BC.

The text corpora for training language models comprise 1.4 billion words, including transcriptions of the acoustic training data, the LDC-released Mandarin Gigaword corpus, GALE-related web text data, and conversational telephone speech text data. A few passes of cleaning are applied to the source text to remove HTML tags, punctuation, corrupted GB2312 codes, and to normalize numbers from digits to spoken forms. Word fragments, laughter, and background noise transcriptions are mapped to a special garbage word.

3.5.4.3 Modeling and Decoding Architecture

Our system uses cross adaptation and system combination between two complementary subsystems. In Zheng *et al.* (2007), it was shown that there were three discriminative techniques that were effective in reducing recognition errors: multi-layer perceptron (MLP) features, minimum phone error (MPE) training criterion, and feature space MPE (fMPE) transform. In practice, we have found that the gains from three discriminative techniques are additive, under a speaker-independent (SI) recognition setup. However, after unsupervised speaker adaptation (SA), combining all three techniques yields minimal further improvement compared with combining only two of

them. Therefore, in designing our two acoustic models (AMs), we decide to choose the most effective combinations of two techniques: MLP+MPE and fMPE+MPE, where the most effective technique, MPE training, is always applied. To ensure diversity in ASR subsystems, we use different cepstral features (MFCC and PLP) for the two systems, and design a new pronunciation phone set for the PLP system with the goal of improving performance on BC speech.

Since Mandarin is a tone language, we have adopted embedded tone modeling technique in both acoustic systems. Tonal phonetic units are used and 3-dimensional smoothed pitch features (Lei *et al.* 2006) are appended to the cepstral features in the front end. For the MFCC system, 32-dimensional MLP features are also appended to the feature vector, resulting in a feature vector of 74-dimension. More details of the MLP features are discussed in by Hwang *et al.* (2009). The PLP system uses 42-dimensional PLP+F0 features, with fMPE transform applied. Except for the Hub4 data where we use the hand-labeled speaker information, automatic speaker clustering is applied to cluster utterances in all other training corpora into pseudo speakers. Then vocal tract length normalization (VTLN) and cepstral mean and variance normalization (CMN/CVN) are applied for each pseudo speaker.

Triphone HMM models are trained first using ML criterion, with 3500 states clustered by decision tree and 128 Gaussian components per state. The Gaussian means are then re-estimated using MPE criterion, with phone lattices generated by running recognition with an unigram LM and ML-trained AM on the training data. The within-word models are used for generating initial hypothesis for adaptation and speaker adaptive training (SAT) based feature transforms. The crossword models are trained with SAT transformed features, and used for decoding in later passes.

BN and BC LMs are trained with modified Kneser-Ney smoothing on the expanded 64K-word decoding vocabulary and interpolated to optimize the perplexity on a heldout set. Pruned bigram and trigram LMs are used in the first pass decoding. The full trigram is used in subsequent lattice rescoring if not applying LM adaptation (LMA), in which case BN and BC component trigram LMs are used. We have also tried using higher order LMs for rescoring but no significant improvement has been achieved.

Figure 3.26 illustrates the flowchart of our recognition system. The audio data are first segmented into small utterances of a few seconds long. Then speaker clustering is applied using Gaussian mixture models of static MFCC features and K-means clustering, same as in training. VTLN is performed for each pseudo speaker, followed by utterance-based CMN/CVN on all features.

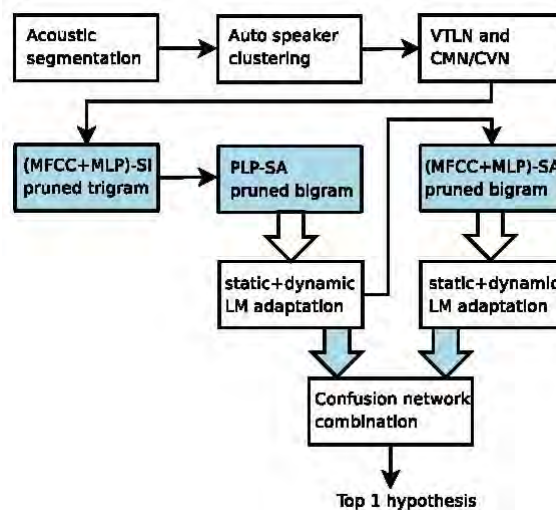


Figure 3.26 System decoding architecture. White block arrows represent lattices and shaded block arrows represent N-best hypotheses.

As shown in the three shaded boxes in Figure 3.26 the decoding consists of three passes:

- 1) (MFCC+MLP)-SI: a speaker-independent within-word tri-phone MPE-trained MFCC+MLP model and the aggressively pruned trigram LM are used for a fast search.
- 2) PLP-SA: next we use the top hypothesis from the last step to learn the speaker-dependent SAT feature transform and perform MLLR adaptation per speaker, on the crossword tri-phone SAT+fMPE MPE trained PLP model. The pruned bigram is used in decoding to generate lattices. Then LM adaptation is performed to rescore the lattices. The top hypothesis and the N-best lists are generated.
- 3) (MFCC+MLP)-SA: Similar to PLP-SA, we first run cross adaptation, using the top hypothesis from the last stage to adapt the crossword triphone SAT MPE-trained (MFCC+MLP) model. The same LM adaptation is performed after the bigram decoding.

Finally, the N-best lists from PLP-SA and (MFCC+MLP)-SA stages are combined using a confusion network to generate the best hypothesis.

3.5.4.4 Data Driven Lexicon Expansion

The decoding lexicon for speech recognition often needs to be updated or expanded over time, especially for Mandarin broadcast speech where new words are constantly being created. We develop a data driven framework to expand the lexicon with multiple pronunciations and new vocabulary words to improve Mandarin ASR performance.

Pronunciation Learning:

A Mandarin word may have different pronunciations depending on context. The pronunciation differences can be due to surface-form variation, such as neutral tones and tone sandhi changes, or because of baseform variation that indicates different meanings. For example, the word “同行” can have the meaning of “walk together” when its tonal Pinyin pronunciation is, “tong2 xing2”, and can also mean “person in the same occupation” when it is pronounced as “tong2 hang2”. Both pronunciation variants are frequent for this word and should be explicitly incorporated in the dictionary as baseforms.

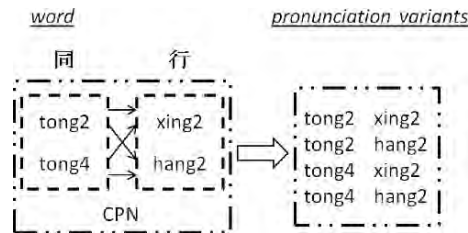


Figure 3.27 Generation of pronunciation variants for a Chinese word from its character pronunciation network (CPN).

As illustrated in Figure 3.27, for a Chinese word, the pronunciation variants of each constituent character are enumerated from a character pronunciation dictionary and combined to construct a character pronunciation network (CPN). Note that “tong4” is a rare pronunciation of the character “同”. The CPN generates all possible pronunciation variants of the word, including neutral tone variants for unstressed characters. After the pronunciation variants are enumerated, tone sandhi rules may be applied to explicitly encode tone changes within the word.

All the variants are then added to the dictionary with equal weight and used to perform forced alignment of the acoustic training data. The recognizer will choose the best pronunciation variant for each word in the sentence. Using the forced alignment statistics, we can select the top pronunciation candidates. If several variants have significant probabilities, these can be selected as multiple pronunciations. Existing dictionary pronunciations can be checked against the estimated pronunciation probabilities to spot dictionary errors.

Vocabulary Expansion:

Extraction of new words or phrases in Chinese has been studied recently in by Zhang *et al.* (2000) and Chang *et al.* (2003). The task is similar to the extraction of multi-words (Finke and Waibel 1997) or compound words (Saon and Padmanabhan 2001) for English ASR. Saon and Padmanabhan (2001) and Chang *et al.* (2003) adopt an iterative approach to learn compound words. In each iteration, the top word pairs are merged according to a language model based measure. The measure is defined as the geometric mean of the *forward bigram* probability $P_f(w_j|w_i)$ between two words w_i and w_j and the *reverse bigram* probability $P_r(w_i|w_j)$

$$\mathcal{LM}_2(w_i, w_j) = \sqrt{P_f(w_j|w_i)P_r(w_i|w_j)} = \frac{N(w_i w_j)}{\sqrt{N(w_i)N(w_j)}} \quad (3.43)$$

where $N(\cdot)$ represents the frequency count of a word in the training data, and $w_i w_j$ denotes the new compound word formed by concatenating two words, w_i and w_j .

The iterative procedure for new word extraction can be time consuming for large corpora, since it requires processing the entire corpus and updating of bigram counts or probabilities at every iteration. For Chinese, it additionally involves word segmentation, and may, therefore, require retraining of the word segmenter. As an efficient shortcut, we can approximate counts for iterated word formations directly from higher order N -gram counts, as follows.

To combine three words w_i , w_j and w_k , we can either combine w_i and w_j first and then combine the new word $w_i w_j$ with w_k , or combine w_j and w_k first and then combine w_i with $w_j w_k$. We use the geometric mean of the merging measures of the two options to create three-word compound words:

$$\mathcal{LM}_3(w_i, w_j, w_k) = \sqrt{\mathcal{LM}_2(w_i w_j, w_k) \mathcal{LM}_2(w_i, w_j w_k)} = \frac{N(w_i w_j w_k w_l)}{\sqrt[4]{N(w_i w_j)N(w_k w_l)}} \quad (3.44)$$

There are many more orderings when combining four words. Again, we could use the geometric mean of merging measures of all possible combination orders. However, since typical Chinese phrases are of length four characters or less, while each word has at least one character, four-word compound words do not appear very often. Hence, we have used the following simplified measure:

$$\mathcal{LM}_4(w_i, w_j, w_k, w_l) = \mathcal{LM}_4(w_i w_j, w_k w_l) = \frac{N(w_i w_j w_k w_l)}{\sqrt{N(w_i w_j)N(w_k w_l)}} \quad (3.45)$$

Based on these merging measures, we can create new compound words with different numbers of constituent words. However, there are cases in which both a longer word $w_i w_j w_k w_l$ and its sub-word $w_i w_j$ are extracted as new words, while the shorter word does not appear by itself and should not be a separate new word. Therefore, after the new words are learned, we need to remove these subset words if their counts are almost the same as those of the corresponding longer words. This is similar to using the context variation criterion, as do Chang *et al.* (2003).

3.5.4.5 LM Adaptation

LM adaptation has been applied successfully to improve the performance of our Mandarin speech recognition system. LM adaptation is performed at the snippet level: for each snippet, the LM is adapted according to the first pass decoding hypothesis to match its topic and genre. The adapted LM is used for rescoreing the bigram lattices. In our system, both static and dynamic LM adaptation are implemented.

In static LM adaptation, a number of static topic LMs are first built to cover different topics. Different from the work of Shi *et al.* (2008), in which a manually constructed topic tree is used, we perform unsupervised clustering on the training corpora using the open source toolkit CLUTO (Karypis 2002) with TF-IDF features. A cosine distance is adopted to measure the similarity between two documents in the training corpora. Bisecting k-way clustering is performed to divide the training corpora into k topic clusters. A static topic LM is trained for each topic cluster. To make the system more up-to-date, we only use data from the most recent ten years (1997-2006) in Gigaword XIN, Gigaword ZBN and Gigaword AFP news corpora to do unsupervised topic clustering. The number of topic clusters is set to 64 (larger number of clusters did not further improve the performance).

Dynamic LM adaptation employs information retrieval technology to search for topic matched documents for each snippet to build LMs. The advantage is that it divides the language model training corpora into finer granularity than static language model adaptation, and thus can build more specific LMs for each snippet. The document ranking algorithm in our system is the traditional vector space model (VSM) for information retrieval. The bigram decoding hypotheses of all the utterances in each snippet are concatenated and used as a query, with stopwords removed. The top n documents with the highest ranking are grouped together to train a dynamic topic LM for the corresponding snippet. In our experiments, n is set to 120. The corpora for retrieval in dynamic LM adaptation include GALE manual transcripts, GALE web data, NTU web data, UW web data and data from the most recent ten years (1997-2006) in Gigaword XIN and ZBN.

For each snippet, we put the 64 static topic trigram LMs and the corresponding dynamic trigram LM together to form a pool of LMs. To reflect the genre variation among different snippets, we also include the component BC LM that is trained with all BC style data. The three LMs with the lowest perplexity on the bigram decoding hypothesis are selected. They are then linearly interpolated with the component BN LM to minimize the perplexity on the corresponding bigram decoding hypothesis.

3.5.4.6 Experimental Results

First, we evaluate the data-driven lexicon expansion approach. We start with a training lexicon of 71K words or so, and select the most frequent 60K words as the decoding lexicon. This 60K-word lexicon is our baseline decoding lexicon, which has several hundred multiple pronunciations. For fast turnaround in experiments, we only perform first pass decoding with the ML trained within-word models, as in (MFCC+MLP)-SI stage shown previously in Figure 3.26.

Pronunciation variants for the 10K most frequent words in the decoding vocabulary are extracted from the expanded CPNs. A single-character dictionary with 6765 characters and 8095 multiple pronunciations is used, averaging 1.2 pronunciation variants per character. After expansion, we replace the pronunciations of these top 10K words in the training dictionary with around 19K generated pronunciation variants. The training data is forced-aligned with an existing AM. From the alignments, we collect the counts of each pronunciation variant and compute their probabilities for each word. If a variant has a probability exceeding 0.9 it is chosen as a single pronunciation. It can be compared to

the pronunciation in the original dictionary to check for a potential error. Otherwise, if a variant has a probability larger than 0.25, it is included as one of multiple pronunciations to be added to the lexicon. The expanded pronunciation lexicon is used to train new acoustic models. The pronunciation probabilities estimated from acoustic training are used to improve recognition by combining them with language model probabilities during decoding. In this experiment, we add 800 pronunciation variants selected by word frequency.

New vocabulary words are extracted from BN and BC transcriptions separately, since the two genres are quite different lexically. About 1600 two-component, 800 three-component, and 100 four-component new words are extracted from both BN and BC data, resulting in a total of 4245 unique new vocabulary items (with less than 20% overlap between BN and BC new words). To generate pronunciations for these new words, the CPN-based pronunciation learning procedure described above is used. As in the multi-word handling of Stolcke *et al.* (2000), the new compound words are used to replace the corresponding constituent words in the text data. The new words are added to the original 60K decoding vocabulary, yielding in a new vocabulary of about 64K. The language model is retrained with the expanded vocabulary.

Lexicon	Dev07			Dev08		
	BN	BC	All	BN	BC	All
baseline	4.9	20.6	13.9	6.2	19.2	12.9
+ pron	4.7	20.1	13.5	5.7	19.0	12.6
+ vocab	4.8	19.4	13.2	5.5	18.3	12.1

Table 3.47: Lexicon expansion CER results (%).

Recognition results in terms of character error rate (CER) are listed in Table 3.47. Adding 800 multiple pronunciations (denoted as “+ pron”) brings significant improvements on both dev07 and dev08. On top of the dictionary with added pronunciations, adding new vocabulary words (denoted as “+ vocab”) achieves 0.7% absolute improvement on the BC subset of both dev07 and dev08. We may attribute this to the fact that conversational speaking style renders words more confusable, and that concatenating words into longer units makes these less confusable again (Saon and Padmanabhan 2001). There are very small improvements on BN subsets after vocabulary expansion. This means the vocabulary expansion is more beneficial for the BC genre, possibly because pronunciations are more affected by co-articulation in that genre. Finally, we have native speakers of Mandarin inspect the added vocabulary items in the final dictionary, and eliminate those that were not semantically plausible as word units. This effort only improves the CER results by 0.1% absolute.

	dev07			dev08		
	BN	BC	All	BN	BC	All
baseline	2.7	14.4	9.4	2.9	13.7	8.5
static only	2.4	14.3	9.2	2.9	13.8	8.6
dynamic only	2.3	14.2	9.1	2.7	13.4	8.3
static + dynamic	2.3	14.2	9.1	2.6	13.5	8.3

Table 3.48: LM adaptation CER results (%).

Then the LM adaptation experiments are performed with our final system for GALE 2008 evaluation (but without RWTH's systems for combination). Table 3.48 shows the results on static and dynamic LM adaptation and their combination. The baseline in the experiments uses the full trigram LM for lattice rescoring. It is shown that the combined LM adaptation can consistently reduce the CER on both LDC dev07 and dev08 set, and on both BN and BC genres.

We also compare the relative contribution of the static and dynamic LM adaptation. Table 3.48 shows that static LM adaptation alone only gives minor improvement on the recognition performance, and dynamic LM adaptation is as good as using static and dynamic LM adaptation together. This confirms our hypothesis that dynamic LM adaptation can provide a finer grained adapted LM for rescoring. There is very little improvement or degradation on the BC genre when using static LM adaptation only. This may be due to that the majority of the text training corpora are in BN genre. The coverage of BC genre is quite limited with static topic LMs. In addition, the relatively higher error rate in BC first pass decoding hypothesis makes it more difficult for either type of LM adaptation.

	dev07			dev08		
	BN	BC	All	BN	BC	All
RWTH only	3.1	13.6	9.2	3.9	13.1	8.7
SRI-UW only	2.3	14.2	9.1	2.6	13.5	8.3
SRI-UW + RWTH						
(MFCC+MLP)-SI	4.0	17.4	11.7	4.9	16.0	10.6
PLP-SA + LMA	2.6	15.0	9.7	3.0	14.2	8.8
(MFCC+MLP)-SA + LMA	2.5	14.6	9.4	2.8	13.9	8.5
combine with RWTH	2.2	12.3	8.0	2.6	12.1	7.5

Table 3.49: System combination CER results (%).

Finally, we perform system combination experiments with RWTH Mandarin STT systems. Two systems from RWTH are used. Each system generates an N-best list for final confusion network based system combination. In addition, the RWTH top-1 hypothesis from combining the two RWTH systems is used for AM adaptation in the PLP-SA stage in Figure 3.26. The performance results of RWTH and SRI-UW systems are shown in Table 3.49. We also show the results of the intermediate stages of the combined system. The two systems are comparable on both dev07 and dev08, yet with different error rates on BN and BC genres. This diversity in error patterns especially helps the combined BC performance. An improvement of 1.1% and 0.8% absolute are obtained from combination on dev07 and dev08, respectively.

3.5.4.7 Conclusions and Future Work

By porting the successful English ASR technologies, and exploiting Mandarin specific characteristics, significant accuracy improvements have been made in our Mandarin recognizer. The more recent work on automatic lexicon expansion and LM adaptation contribute to the performance improvement. The CER performance of our system on BN genre is below 3% on the two development sets. However, although the BN recognition

is highly accurate, the error rate in the BC genre is still high due to its more conversational style. Future work includes investigating acoustic and LM adaptation towards the BC genre to improve BC recognition performance.

Acknowledgements

The authors would like to thank our RWTH colleagues Christian Plahl and Ralf Schlüter for providing the RWTH system output for combination. Thanks also to Dimitra Vergyri and Jing Zheng at SRI for help during the system development.

3.5.5 Improving Speech Transcription for Mandarin-English Translation

Authors: Marcus Tomalin, William Byrne, Adrià de Gispert, Mark J.F. Gales, Xunying Andrew Liu, Phillip C. Woodland, and Kai Yu

3.5.5.1 Introduction

This section presents an examination of the impact of Speech-To-Text (STT) improvements on the performance of a Statistical Machine Translation (SMT) system in the context of a Mandarin-English transcription-translation pipeline. Specifically, the outputs generated by five incrementally improved STT systems are input to a constant CU-SMT system, and the impact of the incremental STT gains is quantified in terms of TER and BLEU scores. In addition, a post-STT Sentence Boundary Detection (SBD) stage is added and this further improves the SMT performance. It is shown that, for this experimental setup, there is an approximately linear relationship between STT and SMT performance improvements.

In recent years, there has been considerable interest in combining STT and SMT technologies as part of the same processing pipeline. If an SMT system takes STT token sequences as input, then it is crucial to determine the impact that decisions made at the STT stage have upon the performance of the SMT component.

This section describes the development of the CU-HTK Mandarin STT system as part of a transcription-translation pipeline, which converts broadcast Mandarin audio data into English text. There are three main components. The STT Component converts Mandarin audio into Mandarin text. The Integration Component post-processes the STT output by mapping spoken numbers to digits and by re-segmenting the token sequences. Finally, the SMT Component translates the post-processed Mandarin text into English.

Five CU-HTK STT systems are discussed in this section and they were all developed as part of the GALE program. They all use a multi-pass, multi-branch framework. Since they use different amounts of data and different configurations, they provide examples of incremental CU-HTK system development. Overall, the improvements gave gains of approximately 45% relative reduction in Character Error Rate (CER) for both Broadcast News (BN) and Broadcast Conversation (BC) data. The STT system outputs were post-processed using an SBD system, which automatically broke the STT segments into sentence-like units. This was done by combining a CART-Style Decision Tree Prosodic Feature Model (PFM) with a 4-gram LM in a lattice-based Viterbi decoding framework.

The re-segmented data was translated into English text using the CU-SMT system. The same CU-SMT system was used for all the experiments. The performance of the complete transcription-translation pipeline was evaluated using the TER and BLEU metrics. It is shown that improving both the STT system and the post-STT segmentations can lower the TER scores by up to 1.7% absolute and increase the BLEU scores by up to 1.0% absolute. Also, it is demonstrated that there is an approximately linear relationship between STT and SMT performance improvements.

3.5.5.2 The CU-HTK Mandarin STT Systems

For the GALE project, the generic CU-HTK architecture was adapted so that it could model Mandarin. The main Mandarin-specific modifications are the following: the front end used pitch-based features (plus delta, delta-delta), which were added after Heteroscedastic Linear Discriminant Analysis (HLDA); the acoustic models used optionally bandwidth-dependent HMMs; tonal questions were added to the decision-tree state clustering process, and a dual-language (Mandarin-English) decoding system was developed. The LMs were 4-gram word-based models which used character to word segmentation based on simple longest match.

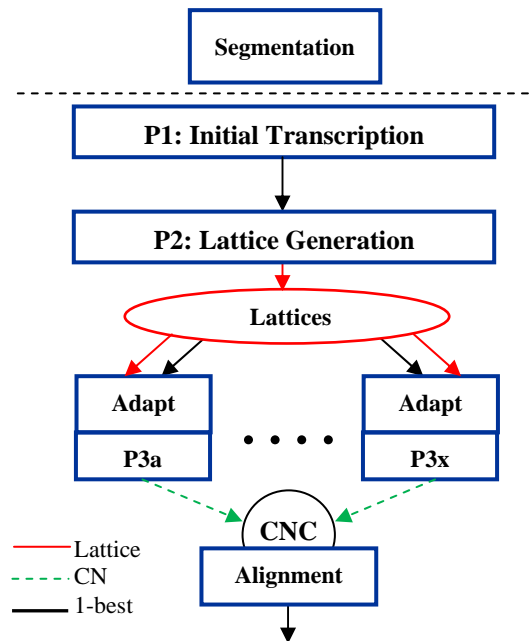


Figure 3.28: The multi-pass and multi-branch framework for the STT-05 and STT-Jan06 systems.

Five versions of the CU-HTK Mandarin STT system were compared. The first system, STT-05, is described by Sinha (2006) and the multi-pass/multi-branch architecture is shown in Figure 3.28. The P1 stage uses gender-independent triphone models trained using Minimum Phone Error (MPE) to provide adaptation supervision for

the P2 lattice generation stage (Povey 2002). The P3 stage performs Constrained Maximum Likelihood Linear Regression (CMLLR) and lattice-based Maximum Likelihood Linear Regression (MLLR) using the 1-best adaptation supervision and lattices from the P2 stage. The final system output was derived by combining P3 outputs from a Feature Minimum Phone Error (fMPE) system which used 46 phones, and a GD 38 phone system which used a Gaussianized frontend. These outputs were combined using Confusion Network Combination (CNC). The acoustic models were trained using 158h of BN audio (including 11h of English data), and MPE trained triphone models were built using this data. The LM was trained on 366M words of data (including the audio data and web-data) and built using a 55k wordlist.

The second system, STT-Jan06, used exactly the same architecture as the STT-05 system. The only difference was that the acoustic models were trained using an additional 19 hours of BC audio data.

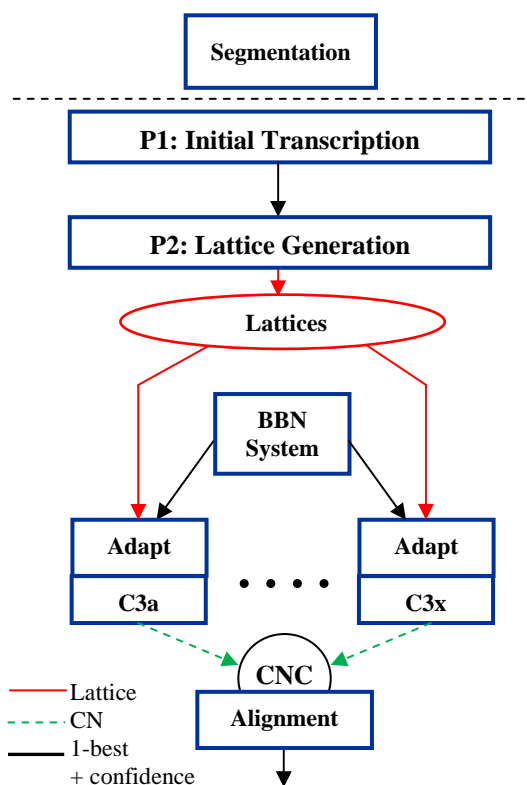


Figure 3.29: The multi-pass and multi-branch evaluation framework for the STT-06 system.

The third system, STT-06, used cross-adaptation rather than 1-best supervision from the P2 stage. Specifically, the CU multi-pass system generated test data lattices, and BBN provided system output as the supervision to adapt the CU acoustic models. The multi-pass BBN system produced an alternative acoustic segmentation (Xiang 2005). The BBN confidence scores for each recognized word were used when adaptation was performed to

the single-best hypothesis. The adapted models were used to rescore the lattices and generate the final output. The STT-06 system architecture is shown in Figure 3.29. The STT-06 system acoustic models were trained on 504 hours of data – the 177 hours used by the STT - Jan06 system, plus an additional 356 hours of which 137 hours consisted of BC data while the rest was BN. Once again, MPE triphones were built. The STT-06 LMs were trained using 1.2G words. The additional LM training data included all the transcriptions of the additional acoustic data, plus downloaded webdata collected at CU up to the end of January 2006, and newswire GALE data. A 3-way interpolation between the BN, BC and Phoenix TV data based component LMs was used to build the final LM, and a tuning set consisting of both BN and BC data was used to optimize the interpolation weights to obtain a more balanced CER performance for both genres. The acoustic and language models used an expanded list of 52k multiple character Chinese words for character to word segmentation. The revised recognition vocabulary contained a total of about 58k words, which includes an additional 5k single-character Chinese words, and 269 frequent English words observed in all the acoustic training transcriptions.

The fourth system, STT-Feb07, also used cross-adaptation from a revised version of the BBN output. The acoustic models were trained on 942 hours of audio (of which 432 hours were BC data), and the LM was trained on 1.3G words. The branches in the P3 stage were MPE SAT, Gender Dependent (GD) MPE, and Gaussianized SAT, and Eval06 models were used for narrowband data processing.

The final system, STT-Dec07 used an additional 1500M words from a range of newswire and web texts. Individual component LMs were built for the 27 sources, and these were interpolated and merged as usual. The wordlist was increased from a 58K to a 63K list (this included 5k of frequently occurring English words). Further, 682 hours of additional GALE BN and BC data were used to train the acoustic models.¹⁰

The five systems all used the same pre-STT segmentation clustering scheme. A Gaussian Mixture Model classifier splits the data into wideband speech, telephone speech, speech with music, and pure music regions. A GD phone recognizer is then run to locate gender-change points and silence, and then a symmetric divergence based change point detector and BIC agglomerative clustering stage is used to refine the segmentation (Sinha 2006). The purpose of this segmentation stage was to divide the audio data into homogeneous acoustic blocks, and a maximum segment length limit of 30 seconds was used.

STT System	bcmdev05			bnmdev06		
	CER	TER	BLEU	CER	TER	BLEU
STT-05	32.9	77.9	10.5	13.4	73.8	15.0
STT-Jan06	25.4	75.3	12.8	11.6	72.8	15.7
STT-06	17.6	73.7	14.3	7.6	72.1	16.4
STT-Feb07	16.8	73.1	14.6	7.3	72.0	16.5
STT-Dec07	15.6	73.4	14.9	6.5	71.8	16.6

Table 3.50: CERs, TERs, and BLEUs, for the five systems for the bcmdev05 and bnmdev06 test sets. The number of segments is constant for each test set (836 for bcmdev05 and 934 for bnmdev06)

¹⁰ Since the Mandarin systems built for the 2008 GALE Evaluation used manual segmentations, these systems were not included in the experiments reported here.

The STT output for two test sets was produced. The `bcmdev05` test set consists of 2.5 hours of Mandarin BC data taken from 5 shows that were broadcast in March 2005. The `bnmdev06` test set contains 3.5 hours of BN Mandarin data, selected from 14 shows that were broadcast between February 2001 and October 2005. The CER results for the five CU-HTK systems are given in Table 3.50. These results show that the incremental improvement in STT performance from the STT-05 to the STT - Dec07 systems produced about a 45% relative reduction in CER for both the `bcmdev05` and `bnmdev06` test sets. The way in which the specific components of these five systems contributed to the CER reduction is a topic that will not be discussed here since the purpose is simply to compare five systems that achieved incremental CER gains. However, it should be noted that roughly 0.5-1.0% absolute of the total CER gain was due to cross-adaptation with BBN. Further information about the CER gains for the systems discussed here can be found in the cited publications.

3.5.5.3 The CU-SMT System

The CU-SMT system used for all the experiments described in this chapter is based on the Transducer Translation Model. This is a phrase-based generative model of translation that applies a series of transformations specified by conditional probability distributions and encoded as Weighted Finite State Transducers (Kumar 2006; Mohri 2002). The system allows for phrase insertions, and the phrase reordering model is implemented by a phrase swap transducer with a flat distribution over all phrases, which allows jumps of up to ten words during decoding for a maximum phrase length of five (Kumar 2005). Minimum Error Training (MET) is applied to adjust the lexical LM scale factor, word and phrase insertion penalties, phrase reordering scale factor, phrase insertion scale factor, forward phrase translation model scale factor, backward phrase translation model scale factor, forward and backward lexical translation models, and three phrase pair count features (Bender 2007).

The CU-SMT system was trained on all data available for the GALE 2008 Mandarin-English Evaluation. The training material contains 10.4M parallel sentences, with roughly 250M words per language, which was automatically word aligned using the MTK package (Deng 2008). The Tune and Test sets are built including non-overlapping sections of various broadcast news and conversation test sets. Each set contains ~60k Mandarin words and one single English reference for SMT scoring. The MET procedure was applied to optimize BLEU on the transcribed Tune set, and results are reported on the Test set. The English language model is an equal-weight interpolation of three 4-gram LMs, estimated over the English side of the parallel text, the AFP and Xinhua portions of Gigaword corpus version 2 (850M words), and an archive of news corpora (825M words), respectively.

3.5.5.4 CU-HTK STT System Performance

The written numbers in the STT output were mapped into digit strings and the output produced by the five systems was translated into English using the CU-SMT system described previously. The translated output was scored using the standard TER and NIST

BLEU metrics. In all the experiments described here, the texts were converted into lower case, and only a single reference was used for the SMT scoring.

CER, TER, and BLEU results are given for the five systems in Table 3.50. The TER results for the `bcmdev05` and `bnmdev06` data show a gradual improvement (with the exception of the `bcmdev05` result for the STT-Dec07 system). While the TERs for the `bcmdev05` are lowered by 4.5% absolute in total, while those for the `bnmdev06` data are lowered by 2.0% absolute. It should be recalled that the CU-SMT system was tuned using BLEU, and this may explain the slightly noisy TER results. However, the same basic asymmetry between the BC and BN results is observed in the BLEU scores: those for the `bcmdev05` data increase by 4.4% absolute while those for the `bnmdev06` data increase by 1.6% absolute. This may simply reflect the fact that the STT systems were improved in part by the addition of BC data. Consequently, the gains for the `bcmdev05` data are greater than those for the `bnmdev06` data.

3.5.5.5 Post-STT Integration

In recent years it has become desirable to produce STT output that contains information concerning sentence boundaries or 'Slash Units' (SUs) (Shriberg 2004; Liu 2005; Tomalin 2006). SUs are 'sentence-like' units, not traditional 'grammatical' sentences. In other words, they do not need to conform to the traditional grammatical definitions of a sentence. The pre-STT segmentation clustering scheme does not provide this kind of segmentation since it merely divides the audio data into homogeneous acoustic blocks. However, there are practical advantages to introducing a post-STT stage which re-segments the STT output into SUs. For instance, there are many important downstream processes and applications that benefit from having SU-segmented STT output. Since SMT systems can function as downstream components in a transcription-translation pipeline, it is useful to quantify the extent to which they benefit from taking sentence-like STT segmentations as input.

The post-STT segmentation strategy explored in these experiments takes automatically segmented transcriptions of Mandarin audio as input and splits the STT segments into smaller units using sentence boundary information. Crucially, only the segment boundaries are changed; the token sequences remain the same. The guiding assumption is that the pre-STT segmenter under segments the data, and therefore, a subsequent segment splitting stage is beneficial. These sentence-like segments are then passed to the SMT component to be translated into English text. The post-processing SBD scheme used here is PFM-NGRAM. This system combines a CART-Style Decision Tree Prosodic Feature Model (PFM) and a 4-gram in a 1-Best Viterbi decoding framework. It is based on the system discussed by Tomalin (2006). Specifically, a word-based 4-gram was constructed using 250M words of LM training data that had been marked with a binary sentence-boundary token. The PFM was built using pause and duration features that had been extracted from 28 hours of acoustic BN data. The PFM was created as described by Tomalin (2006). The 4-gram was combined with the PFM in a lattice-based 1-Best Viterbi decoding framework with an empirically determined grammar scale factor of 0.8. The 1-Best decoder output produced token sequences for

each file in the test sets, and these contained the STT lexeme token sequence and SU boundary tokens that had been inserted automatically during the decoding process.

Test	SBD	#Segs	CER	TER	BLEU
bcmdev05	×	836	15.6	73.4	14.9
	√	2197	15.6	71.7	15.6
bnmdev06	×	934	6.5	71.8	16.6
	√	2240	6.5	70.6	17.6

Table 3.51: CERs, TERs, and BLEUs, for the STT-Dec07 + SBD output for the bcmdev05 and bnmdev06 test sets.

The bcmdev05 and bnmdev06 STT outputs were post-processed using the PFM-NGRAM scheme and the performance of the five composite transcription-translation systems was evaluated. The detailed results for both the STT - Dec07 and STT - Dec07+SBD systems are given in Table 3.51. These results show that the PFM-NGRAM scheme generate more than double the number of pre-STT segments. Crucially, though, the STT-Dec07+SBD system obtains lower TERs and higher BLEU for both test sets (up to 1.7% absolute gain in TER and up to 1.0% absolute gain in BLEU). This quantifies the extent to which the CU-SMT system benefits from taking post-STT SBD output as its input, rather than the unsegmented STT output.

To examine the relationship between the STT and SMT components, the CER scores were plotted against the TER and BLEU values for all systems. The plots are given in Figure 3.30. These results confirm that the CU-SMT system consistently produces better TER and BLEU scores when it uses the STT+SBD output rather than the unsegmented STT output. More importantly, the CER improvements for the bcmdev05 data produce both TER and BLEU gains for the same SMT system, and the relationship between the performance of the STT and SMT systems is roughly linear whether post-STT SBD processing is applied or not. Crucially, this pattern is consistent across different data genres, since the same general trend is observed both for the BC and for the BN test sets. The patterns are not quite so clean for the TER results, but, as mentioned above, this may simply be due to the fact that the SMT system had been tuned using BLEU.

3.5.5.6 Conclusions

A transcription-translation pipeline which converts Mandarin audio into English text has been explored. Five CU-HTK STT systems have been contrasted, and it has been shown that the overall improvement was a 45% relative reduction in CER for both BN and BC data. When TER and BLEU scores are obtained for the five different STT outputs, the numbers show an incremental performance improvement. The use of an SBD stage of post-STT processing gives further gains. Most importantly, the CER, BLEU, and TER results for the five combined STT+SMT systems show that (at least for this set-up) there is a roughly linear relationship between the STT and SMT improvements.

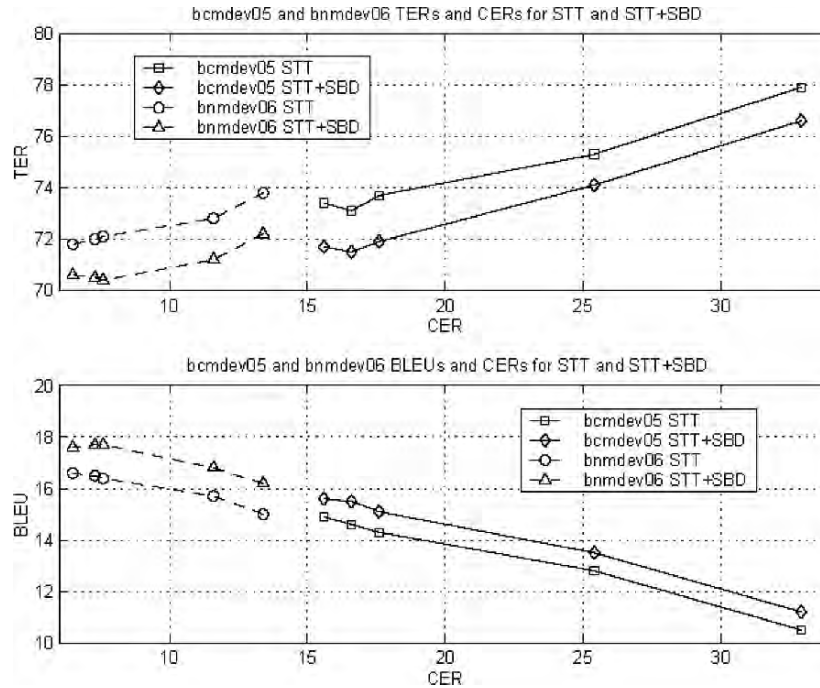


Figure 3.30: CER, TER, and BLEU scores for the bcmdev05 and bnmdev06 data

3.6 Language-Specific Models and Systems: Arabic

3.6.1. Introduction

Author: Katrin Kirchhoff

The Arabic language poses unique challenges for both automatic speech recognition and machine translation, including a highly ambiguous writing system, dialectal variation, and morphological complexity. The resulting problems for automatic speech recognition (ASR), particularly as a frontend to a machine translation module, are briefly summarized below.

Writing System/Acoustic and Pronunciation Modeling:

Arabic is written using a script-based alphabet consisting of 28 letters, 25 of which represent consonants and 3 of which represent the long vowels /i:/, /a:/, and /u:/. Letters may have up to four different shapes, depending on their position in the word. Short vowels are not represented by letters of the alphabet, but are marked by so-called *diacritics*, short strokes placed either above or below the preceding consonant. Several other pronunciation phenomena are marked by diacritics, such as consonant doubling

(phonemic in Arabic), which is indicated by the “shadda” sign, absence of a vowel (marked by the “sukuun”), and the “tanween”, word-final adverbial markers which add /n/ to the pronunciation. Text is normally undiacritized and diacritics are only used in ambiguous contexts or for special text types. While the alphabet has an almost one-to-one correspondence with phones, the absence of diacritics introduces a high degree of phonetic ambiguity. An undiacritized form can have dozens of possible diacritized forms.

Training data for speech recognition is mostly undiacritized since the manual annotation of diacritics is error prone and time consuming. While limited amounts of diacritized text may be obtained for model bootstrapping or evaluation purposes, the vast majority of training data is deficient in the phonetic information it provides. Even though Arabic speech recognizers typically output undiacritized forms, it has been shown in the past that modeling the phonetic information conveyed by diacritics system internally improves recognition performance (Kirchhoff 2003; Afify 2005).

The main research question is how to model the missing phonetic information in the acoustic and pronunciation modeling components of the recognizer. The most frequently pursued approach is to automatically identify the correct diacritized form for each undiacritized base form and to use the diacritized training data for acoustic model training. A frequently used tool in this context is the Buckwalter morphological analyzer (Buckwalter 2002) that provides all possible morphological analyses of an Arabic word along with its possible diacritizations. A forced alignment of the speech signal and training transcriptions enriched with the Buckwalter variants can then be performed. The variants chosen during forced alignment can then be included in the pronunciation dictionary. Virtually all GALE systems use this approach. The common finding is that this procedure results in approximately three to four pronunciation variants per undiacritized word form on average, and that enriching variants with probabilities (based on counts in the training data) during recognition improves performance.

However, there is usually some percentage of words (around 5 - 10%) that cannot be processed due to spelling errors or dialectal origin of the words. Systems differ as to how these words are handled. The IBM system (Soltau *et al.* 2007) simply backs off to unvowelized forms. Lamel *et al.* (LIMSI) describe an approach where forms not handled by the Buckwalter Analyzer are expanded to include a either single generic short vowel or no vowel at all in every possible position. This allows for a better match between training transcriptions and acoustic signal, but avoids the effort involved in predicting exact vowelizations, or the explosion of pronunciation variants caused by including all three short vowels. A comparison of systems where 15% of the training lexicon had the correct vowel vs. the generic vowel showed a 5% relative degradation on broadcast news data, but only 1% on broadcast conversation data. Finally, automatic grapheme-to-phoneme mapping can be applied to generate pronunciation variants for unvowelized forms. The section by Bisani and Ney (RWTH Aachen) describes a statistical joint sequence model to perform this task, which achieved a phoneme prediction error rate of roughly 15% and was used in the SRI Arabic recognition system. Since the different approaches (back off to unvowelized form, generic vowel, phoneme prediction) were evaluated on different data sets it is not possible to compare the results; moreover, their performance is strongly influenced by the overall design of the recognition system.

Several systems (Nallasamy *et al.*, Soltau *et al.*) use both vowelized and unvowelized models and combine the results obtained by systems with different models, or they perform cross-adaptation. Lamel *et al.* specifically studied the effect of diacritics other than vowels, i.e., shadda and tanween, and found that using dedicated phone models for these slightly improved performance.

Dialectal Variation:

Arabic is not a single homogeneous language, but a collection of varieties comprising a written language standard, Modern Standard Arabic (MSA), as well as, a multitude of regional dialects, e.g., Egyptian Arabic, Levantine Arabic, or Gulf Arabic. Whereas MSA is primarily used for written communication and formal discourse, dialectal Arabic is the variety of choice for everyday conversation. Dialects differ from MSA at all linguistic levels and may be used side-by-side with MSA, resulting in code switching even within individual utterances. Dialectal speech can influence recognition performance by causing the misrecognition of dialectal words due to gaps in the recognizer vocabulary, or due to mismatches in acoustic or pronunciation models caused by dialectal vowel changes or omissions.

The default approach to modeling dialectal characteristics is to simply train acoustic models on mixed MSA and dialectal data in the hope that sufficiently powerful acoustic models (e.g., with a large enough number of parameters) will be able to accommodate the additional variation introduced by dialectal phonetic forms. Of the sections in this section, only Soltau *et al.* (IBM) describe an explicit modeling approach, where dialectal questions were included in the decision trees used for clustering acoustic models, resulting in consistent word error rate improvements across different evaluation conditions.

Nguyen *et al.* (BBN) mention including dialectal affixes into their pronunciation dictionary, but the effect on system performance has not been reported. To date, comparatively little work seems to have been performed on dialectal modeling and future work in this area can be expected to show additional gains.

Morphological Complexity:

Arabic is characterized by rich and productive morphological processes. Arabic open-class words consist of a consonantal root interspersed with different vowel patterns, plus various derivational affixes and particles. A given root form can lead to thousands of different word forms, which may present a problem for language modeling and vocabulary selection. In ASR, there are generally two ways to handle this problem: (1) decomposition of the word form into smaller units, or (2) increasing the vocabulary of the speech recognizer as much as possible. Earlier work on Arabic speech recognition, particularly dialectal speech recognition, only had limited training resources available and, therefore, performed word decomposition to improve recognition performance (Vergyri 2004; Choueiter 2006).

Recent advances in efficient decoder design, as well as, the availability of large amounts of (MSA) training data, have permitted systems to simply include all desired word forms as separate entries in the vocabulary, and most GALE systems pursue this approach now. The vocabulary sizes range between 240k and 740k. However, Nguyen *et*

al. (BBN) explicitly compare recognition systems based on words as recognition (language modeling) units with systems based on two different morphemic decompositions and demonstrate that a system based on a linguistic decomposition into morphemes outperforms a standard word-based system in terms of both out-of-vocabulary rate and word error rate. At the same time, the morpheme-based systems have significantly fewer parameters due to smaller vocabulary sizes and language models. The combination of word-based and morpheme-based systems further improves performance.

The conclusion is that Arabic morphology does not present a genuine problem for ASR anymore (provided that a sufficient amount of training data is available). Unlike highly agglutinative languages like Finnish or Estonian, Arabic can be handled by a standard system based on words as recognition units. Careful morphological modeling can yield improvements over the simple word-based approach while yielding a smaller and faster system; however, more linguistic resources (such as morphological analyzers) are required for this approach.

3.6.2. Explicit and Implicit Modeling of Short Vowels for Arabic STT

Authors: Lori Lamel, Abdelkhalek Messaoudi and Jean-Luc Gauvain

3.6.2.1 Introduction

Since Arabic texts are almost always written without diacritics, the main challenge in pronunciation modeling is to deal with incomplete information. A strongly consonantal language with nominally only three vowels, Arabic is a highly inflected language. There are typically multiple possible vocalizations for a given written word. Thus, one of the challenges of explicitly modeling vowels in Arabic is to obtain vocalized resources, or to develop efficient ways to use non-vocalized data (Vergyri and Kirchhoff 2004). Given a properly vocalized form, Arabic grapheme-to-phoneme conversion is (relatively) straightforward. The Buckwalter morphological analyzer (Buckwalter 2002) is widely used to produce possible vocalizations. However, not all words are successfully processed by the Buckwalter Morphological Analyzer, which mainly fails on proper names, technical words, and words in Arabic dialects. In the case of a large quantity of training data, there can easily be many words without vocalizations and determining vocalizations for these words manually or even semi-automatically can be prohibitive. Even though most Arabic speech recognizers output a non-vocalized word sequence, it has been shown that better recognition performance is obtained by explicitly modeling short vowels (Afify *et al.* 2005) than with a grapheme-based approach (Billa *et al.* 2002).

Therefore, in order to simplify the problem of training on non-vocalized audio data, a generic vowel model has been introduced to hold the place of short vowels. Rules were developed to automatically generate pronunciations from the non-vocalized word form. After applying the rules, each word has multiple pronunciations represented with consonants, long vowels, and the generic vowel. The absence of a vowel (the Sukoun) is modeled by removing a generic vowel. In addition to the explicit modeling of short vowels, extending the phone set to explicitly model gemination was found to improve recognition performance. The ‘tanwin’ is another grammatical mark which specifies that

a noun is non-definite. The tanwin causes short vowels in word final position to be 'doubled', which is phonetically realized as adding an 'n' after the final vowel (also called nunation). As for the short vowels, the gemination and tanwin markers may be useful for translation or other downstream processing.

While the original purpose of the generic vowel models was to simplify training, in a second phase, the generic vowel was also used in the recognition lexicon. This allowed the inclusion in the lexicon of words that were previously excluded since they were not handled by Buckwalter. Including them was found to improve lexical coverage and reduce perplexity of development data.

3.6.2.2 Generating Pronunciations with Generic Vowels

One possibility considered to process words that were not able to be processed with the Buckwalter Morphological Analyzer, was to allow all three short vowels or no vowel after every consonant. This idea was quickly rejected since there are too many possible vocalized forms. For example, with words with 4 consonants generate 512 possible pronunciations, and words with 8 consonants have 8192 possible pronunciations. In order to reduce the combinatorial explosion, a generic vowel was proposed to serve as a place holder for any of the three short vowels (Lamel *et al.* 2007). This does not pose any problem since even though short vowels are represented internally in the system, the Arabic recognizer outputs the non-vocalized word form.

Using a generic vowel offers two main advantages. First, the manual work in dealing with words that are not handled by the Buckwalter Morphological Analyzer. These words can be automatically processed. Second, the number of vocalizations, and hence pronunciations, per word is greatly reduced (one vowel instead of three).

A set of detailed rules were used to generate pronunciations with a generic vowel from the non-vocalized word form. Some rules concern the word initial Alif (support of the Hamza), which can be stable or unstable. For the former case a pronunciation is generated with a glottal attack (denoted /'/) followed by a generic vowel (denoted /@/). These rules also cover word initial letter sequences [wAl, wbAl, wkAl, fAl, fbAl, fkAl], which often correspond to a composed prefix ending in "Al". Different pronunciations are generated to represent both situations. For example, the possible pronunciations for wAl are: /w@l/, /w'@l/, and /wAl/. In word final position, short vowels can be followed by an "n" (tanwin), so two forms are proposed, the generic vowel alone and the generic vowel followed by an "n".

Specific rules handle the pronunciation of words ending in "wA" and a final letter "p" (which symbolizes the ta marbouta). Within a word, a generic vowel is added after each consonant with the exception of the semivowels "w" and "y", which can be realized as semivowels or can serve as a support for the long vowels /U/ and /I/, respectively. Similarly a word internal Alif can represent the long vowel /A/ or a glottal attack.

After applying these rules, each word has multiple pronunciations represented with consonants, long vowels, and the generic vowel. Since vowels may also be absent (written with a Sukoun), additional pronunciations are generated by removing one generic vowel at a time. Using the rules the following two generic vowel forms are generated for the word "ktb".

ktb k@t@b@ k@t@b@n

which after allowing each generic vowel to be deleted produces:

ktb k@t@b@ k@t@b@n
 kt@b@ kt@b@n
 k@tb@ k@tb@n k@t@b

These generic vowel rules generate a large number of pronunciations per word: on average 35, compared to 6 per word with a vocalized form.

A test was done to assess the impact of using a generic model instead of three short vowels. All short vowels in the vocalized lexicon were mapped to a generic vowel (@), and training context dependent (CD) models with the standard consonant set and the single generic vowel. A pronunciation lexicon was created using the standard pronunciations with short vowels for the vocalized words, and automatically generated pronunciations with the generic vowel for the non-vocalized words. The audio data was then segmented using this lexicon with a combined set of acoustic models formed by merging the CD models with short vowels and those with a generic vowel. Note that since the basic idea was to use the generic vowel only in training, and not during recognition, a number of CD models are never used.

Two model sets were built with 5k tied states (64 Gaussians per state) and covering 5k phone contexts, one modeling 37 phones and the other 35 (the three short vowels being replaced by a single one). These models were tested using a single pass system without acoustic model adaptation. On broadcast news data (bnat06) there was about a 5% relative performance degradation using only the generic vowel. For broadcast conversation (bcat06) data, the relative loss was less than 1%. The smaller degradation on broadcast conversations may be due to a higher proportion of dialect or less formal data, for which there is a larger variability in the way that vowels are pronounced. These results provided an indication the generic vowel could be an effective means for facilitating training on non-vocalized data. All acoustic models used in the LIMSI GALE systems were built using dictionaries for which about 15% of the words have generic vowels in their pronunciation (Lamel *et al.* 2007).

3.6.2.3 Generic Vowels in the STT Lexicon

Since the generic vowel was found to be helpful in acoustic model training, a natural second step is to also use it in the recognition lexicon. When typical frequency-based methods are used select a recognition word list, a number of words may be excluded if no pronunciation is readily available. For example, when selecting a 200k word list, 36k words that should have been included, were excluded since they could not be vocalized by Buckwalter. As for the training data, these words were mostly proper names, technical terms or words in various Arabic dialects. Adding these 36k words to the 200k word list reduced the OOV rate by 0.2 to 0.9% absolute for different development data sets, showing the interest of including them. A simplified method for generating a limited number of vocalized forms (and thus pronunciations) for recognition was developed. The

basic idea is to put a generic vowel after each consonant and to map all semivowels to a long vowel. This corresponds to the manner in which foreign names and technical terms are spoken. The distribution of the number of pronunciations per word automatically generated in this manner is shown in Table 3.52. About 10% of the words have a single pronunciation, and over 50% have only one or two. Only 2% of the words have more than 10 forms, with the maximum being 16, and all of these words (1.7%) include the prefix 'Al' Using the extended 236k wordlist reduced the word error rate by 0.2% absolute using a single decoding pass without cluster-based adaptation on the eval06 data set. In the reference transcripts, there were 162 occurrences of one of the 36k word forms, accounting for 0.6% of all word tokens. Thus, one third of the possible gain was obtained.

.#pronunciations	% words
1	9.6%
2	42.7%
3	20.4%
≤10	98.0%
16	1.7%

Table 3.52: Distribution of the number of pronunciations with generic vowels for the 36k words in the extended 236k word list.

3.6.2.4 Modeling Geminate and Tanwin

The initial phone set for Arabic contained 37 symbols: 28 Arabic consonants, 3 foreign consonants, and 6 vowels (i,a,u short and long) (Messaoudi *et al.* 2005). When pronunciations were generated with this phone set, all consonants with a gemination mark were simply doubled. While this may be a reasonable approximation for some sounds, such as fricatives, it is clearly not well adapted to plosives where gemination does not result in multiple bursts. Other sources of gemination arise when a contextual variant of the definite article 'Al' ('the'). When the 'Al' precedes a lunar consonant it is usually pronounced as /al/. When the 'Al' precedes a solar consonant it is usually silent, but transforms the following consonant into a geminate. Generally speaking all of the Arabic consonants can occur as singletons or geminates.

The left part of Figure 3.31 illustrates a portion of the phrase “(kaAn)ati AIT aAgirap Al\$ ir(aAEiyap)”. An aligned approximate phone transcription is shown on the bottom. The sukoun indicating that the vowel following the first /t/ is not pronounced is transformed to a short /i/, because it precedes an 'Al'. The 'Al' in turn precedes a solar consonant so it is not pronounced, but causes the T (emphatic t) to be geminated. The short /i/ (around time 783.8) is realized as a schwa-like vowel. Another geminate 'sh' (SS) is centered at time 784.4.

In order to more precisely model such phenomena, an alternative phone set was explored in which an additional 30 phone symbols were added to represent the geminate phones. The frequencies of the consonants in single and geminate form were counted in a 100 hour corpus of manually transcribed and vocalized Arabic broadcast news data (Messaoudi *et al.* 2005). The right part of Figure 3.31 lists the solar and lunar consonants, along with the percentage of occurrences as geminates. It can be observed that the solar consonants generally have a higher proportion of geminates than the lunar ones.

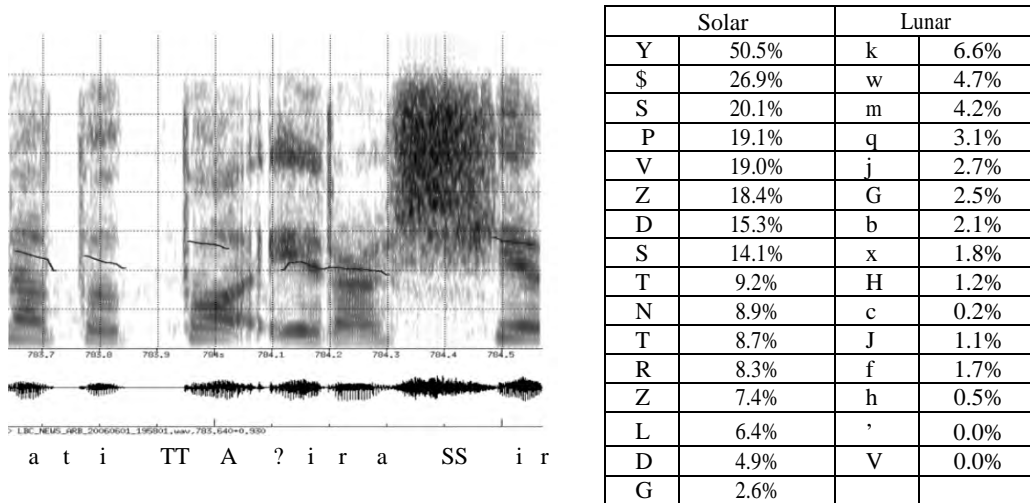


Figure 3.31: **Left:** Spectrogram illustrating gemination. **Right:** Occurrences (%) of geminates for Solar and Lunar consonants.

Figure 3.32 shows how the geminates are represented in the original pronunciation dictionary (top) and the dictionary with specific geminate symbols. The left column gives the transliterated lexical entry. Associated with each entry are several pairs of vocalized forms and their pronunciations.

To compare the original phone set and the extended one with geminates, acoustic models were trained on a corpus of about 1000 hours Arabic broadcast data using the both. They were tested on the broadcast news (bnat06) and broadcast conversation (bcat06) data, each containing about three hours of audio data (Lamel *et al.* 2007). For both data types using specific phones to model geminates gave a small performance improvement, and a further gain was obtained by combining the two models. Increasing the phone set also has the advantage of increasing the number of context-dependent phones that are modeled.

Lexical Entry	Pronunciation = Transliteration			
ktAb	kitAb= kitaAb	kitAba= kitaAba	kuttAb= kut~aAb	kuttAba= kut~aAba
	kitAbi= kitaAbi	kitAbin= kitaAbK	kuttAbi= kut~aAbi	kuttAbin= kut~aAbK
	kitAbu= kitaAbu	kitAbun= kitaAbN	kuttAbu= kut~aAbu	kuttAbun= kut~aAbN
ktAb	kitAb	kitAba	ku+Ab	ku+Aba
(with	kitAbi	kitAbin	ku+Abi	ku+Abin
geminates)	kitAbu	kitAbun	ku+Abu	ku+Abun

Figure 3.32: Sample pronunciations for ktb in the dictionary without (top) and with (bottom) geminate symbols. The lexical entry is the non-vocalized word class encompassing all possible vocalized forms. On the left of the equal sign is the phonemic form and with the corresponding transliterated graphemic form on the right. The transliterated form is not repeated in the bottom part of the figure.

The 'tanwin' is a grammatical mark which specifies that a noun is non-definite. The tanwin causes short vowels in word final position to be 'doubled', which is phonetically

realized as adding an 'n' after the final vowel (also called nunation). The tanwin can be realized as a vowel-n sequence or a nasalized vowel, or some combination. In an attempt to better capture this variability, three phones were added to the phone set representing the three tanwin phones (in, an, un) with a single unit. Acoustic models were built using this new phone set, and tested on the development data sets.

Comparable word error rates were obtained with these models and the non-tanwin models, and an absolute gain of 0.4% was achieved with system combination. Contrastive models were built permitting multiple forms for tanwin, both as a single phone and as a short vowel-n sequence. These models had comparable performance as the previous model sets. Given the large variability in the realization of tanwin, these results are not surprising and suggest that any reasonable, coherent representation for tanwin will suffice.

3.6.2.5 Conclusions

This section has summarized recent advances in pronunciation and acoustic modeling to explicitly take into account some of the specificities of the Arabic language. One of the challenges is training with incomplete information since most Arabic texts are written without diacritics, yet the diacritics provide useful information for pronunciation modeling and higher level processing. Many vocalized word forms can be derived using the Buckwalter morphological analyzer and modifications thereof. In order to enable training for words that Buckwalter is not able to process, rules to generate pronunciations with a generic vowel have been proposed. This method has been used to significantly facilitate training on non-vocalized data, and all of the acoustic models used in the LIMSI GALE systems were trained with them. Pronunciations for 16% of the words in the training dictionary and 11% in the recognition dictionary have a generic vowel. Performance gains were also obtained by explicit modeling of geminate consonants and pronunciation variants of the definite article 'Al'. Some other variants have also improved performance, such as modeling the g-sound in Egyptian Arabic, the tendency for a final /a/ to be realized as a front vowel /i/ in Lebanese data, and the trend to have a Sukoun as the final vowel in spoken Arabic, in particular for dialectal speech.

3.6.3. Statistical Modeling of Pronunciation Lexica

Authors: Maximilian Bisani and Hermann Ney

3.6.3.1 Introduction

Today's speech recognition systems are able to recognize arbitrary sentences over a large, but finite vocabulary. Two key components of such a system are the language model, which describes how likely sequences of words are, and the acoustic model which estimates the likelihood of a string of phonemic symbols given the observed acoustic signal. Both these models are trained using large amounts of text and audio data respectively. They are linked by the pronunciation lexicon, which provides phonemic transcriptions for each word in the vocabulary. The lexicon is not commonly derived from example data, but handcrafted by human experts, which makes it an expensive and often limited resource.

Most application domains require very large lexica to achieve satisfactory coverage. Typically, on the order of one hundred thousand word forms are needed for English. Most languages have a richer morphology than English and require even larger vocabularies. Many important speech recognition tasks feature an open, constantly changing vocabulary, e.g., broadcast news transcription. Extending the vocabulary of a speech recognition system requires, again, human expertise in phonetics.

In alphabetic writing systems the orthographic form is a conventional representation of a word's pronunciation, which makes it possible in principle to convert the grapheme sequence into a phoneme sequence, albeit some margin of error needs to be accepted. The difficulty of this conversion problem varies from language to language and so does the accuracy achieved on this problem by state-of-the-art techniques. Knowledge-based approaches to implement this conversion are rule based pronunciation converters, which need to be developed by phonetic experts. A very simple, straight forward approach is to use the written form directly as a graphemic representation of the pronunciation of a word. The task of learning the relation between a letter in context and its corresponding acoustic realization is simply delegated to the acoustic model. While this has been shown to work reasonably well (Kanthak and Ney 2001), for many languages, a manually created phonetic pronunciation lexicon outperforms this naive graphemic pronunciation approach widely.

A compromise between the knowledge-based and the knowledge-free approach is data-driven grapheme-to-phoneme conversion, where a model of the letter-to-sound relation is derived automatically from an existing dictionary of example pronunciations. Here, we present the results of using such a language independent, fully data-driven grapheme-to-phoneme converter, which can be used as a statistical pronunciation lexicon. A joint multigram model for pairs of grapheme and phoneme sequences is trained on a source pronunciation lexicon. The model can subsequently be used to generate pronunciations and estimate pronunciation probabilities for arbitrary words.

This approach, the software implementation of which is available under an Open Source license, has been successfully applied to different languages and tasks:¹¹

- Offline automatic extension of the pronunciation lexicon (lexicon augmentation) of the Arabic Nightingale transcription system (Vergyri *et al.* 2008).
- Low-cost generation and validation of pronunciation lexica (Bisani and Ney 2008).
- Implicit, online automatic extension of the recognition vocabulary on the basis of sub-word units derived from grapheme-to-phoneme conversion (open vocabulary speech recognition (Bisani and Ney 2005), optionally supported by morphological decomposition).

¹¹ <http://www-i6.informatik.rwth-aachen.de/web/Software/g2p.html>

3.6.3.2 Joint Sequence Models

Statistical problem formulation

An orthographic form is given as a sequence of letters, also referred to as characters or graphemes. We denote the set of graphemes as G . A pronunciation is represented in terms of a phonemic transcription, i.e. a sequence of phoneme symbols. The set of phonemes is denoted as Φ . The task of grapheme-to-phoneme conversion, or phonetic transcription, can be formalized using Bayes' decision rule as:

$$\varphi(g) = \operatorname{argmax}_{\varphi' \in \Phi^*} p(g, \varphi') \quad (3.46)$$

This means, for a given orthographic form $g \in G^*$ we seek the most likely pronunciation $\varphi \in \Phi^*$. Here V^* denotes the set of all strings over symbols in V (Kleene star). It should be noted that this decision strategy is optimal with respect to word error, i.e., the risk of not getting the correct pronunciation is minimized.

Co-segmentations and graphones

The fundamental idea of joint-sequence models is that the relation of input and output sequences can be generated from a common sequence of joint units which carry both input and output symbols. In the simplest case, each unit carries zero or one input and zero or one output symbol. This corresponds to the conventional definition of finite state transducers (FST). When units may carry multiple input and output symbols, the terms co-sequence and joint multi-gram (Deligne *et al.* 1995) are used. However, it has been shown that already the simple FST-like model yields excellent accuracy on the grapheme-to-phoneme conversion task (Bisani and Ney 2008). Chen (2003) reports very good results with a structurally similar model, a joint maximum entropy M-gram model. We describe the simple FST-like model in the following.

We call a pair $q = (g, \varphi) \in Q = (G \cup \{\epsilon\}) \times (\Phi \cup \{\epsilon\})$ of a letter and a phoneme, where either of which can be empty, a singular graphone. ("Singular" because of the restriction to at most one symbol on each side.) The expressions g_q and φ_q refer to the first and second component of q respectively. For example, the pronunciation of "mixing" may be regarded as a sequence of seven graphones:

"mixing" =	m	i	x	—	i	n	g
[mIksIN]	[m]	[I]	[k]	[s]	[I]	—	[N]

Since there are multiple ways to align a given grapheme and phoneme sequence the joint probability $p(g, \varphi)$ is determined by summing over all matching graphone sequences:

$$p(g, \varphi) = \sum_{q \in S(g, \varphi)} p(q) \quad (3.47)$$

where $q \in Q^*$ is a sequence of graphones and $S(g, \varphi)$ is the set of all co-segmentations of g and φ :

$$S(\mathbf{g}, \boldsymbol{\varphi}) := \left\{ \mathbf{q} \in Q^* \left| \begin{array}{l} \mathbf{g}_{q_1} \parallel \dots \parallel \mathbf{g}_{q_K} = \mathbf{g} \\ \boldsymbol{\varphi}_{q_1} \parallel \dots \parallel \boldsymbol{\varphi}_{q_K} = \boldsymbol{\varphi} \end{array} \right. \right\} \quad (3.48)$$

Here \parallel denotes sequence concatenation and $K = |\mathbf{q}|$ is the length of the graphone sequence \mathbf{q} . The joint probability distribution $p(\mathbf{g}, \boldsymbol{\varphi})$ has thus been reduced to a probability distribution $p(\mathbf{q})$ over graphone sequences $\mathbf{q} = q_1, \dots, q_K$, which can be modeled using a standard M-gram approximation:

$$p(q_1^K) \cong \prod_{j=1}^{K+1} p(q_j | q_{j-1}, \dots, q_{j-M+1}) \quad (3.49)$$

Positions $i < 1$ and $i > K$ are understood to hold a special boundary symbol $q_i = \perp$ which allows modeling of characteristic phenomena at word starts and ends.

Model estimation

Given is a training sample of N words paired with their pronunciations $\mathcal{O}_1, \dots, \mathcal{O}_N = (\mathbf{g}_1, \boldsymbol{\varphi}_1), \dots, (\mathbf{g}_N, \boldsymbol{\varphi}_N)$, but without an alignment on the level of letters and phonemes. First, we note that if we have a joint sequence model we can compute the probability of any co-segmentation for each sample, since a co-segmentation \mathcal{S} uniquely defines a joint sequence:

$$p(\mathbf{g}, \boldsymbol{\varphi}, \mathcal{S}) = p(\mathbf{q}) \quad (3.50)$$

Thus, the log-likelihood of the training data can be formulated as the sum over all segmentations:

$$\log \mathcal{L}(\mathcal{O}_1, \dots, \mathcal{O}_N) = \sum_{i=1}^N \log \mathcal{L}(\mathcal{O}_i) = \sum_{i=1}^N \log \left(\sum_{\mathcal{S} \in \mathcal{S}(\mathcal{O}_i)} p(\mathcal{O}_i, \mathcal{S}) \right)$$

The segmentation \mathcal{S} into joint units is a hidden variable. As first demonstrated by Deligne and Bimbot (1995), maximum likelihood training of this model can be performed using the expectation maximization (EM) algorithm. To start the iterative procedure we initialize the unigram model with a flat probability distribution, i.e., all singular graphones have the same initial probability. Higher order M-gram models are initialized using the previously generated (M-1)-gram model.

As previously proposed (Bisani and Ney 2008), we use a discounted re-estimation equations, using M absolute discounting parameters d_1, \dots, d_M :

$$p(\mathbf{q}; \boldsymbol{\vartheta}) = \prod_{j=1}^{|\mathbf{q}|} PM(q_j | h_j; \boldsymbol{\vartheta}) \quad (3.51)$$

$$e(q, h; \boldsymbol{\vartheta}) := \sum_{i=1}^N \sum_{q \in S(g_i, \varphi_i)} p(\mathbf{q} | \mathbf{g}_i, \boldsymbol{\varphi}_i; \boldsymbol{\vartheta}) n_{q,h}(\mathbf{q}) = \sum_{i=1}^N \sum_{q \in S(g_i, \varphi_i)} \frac{p(\mathbf{q}; \boldsymbol{\vartheta})}{\sum_{q' \in S(g_i, \varphi_i)} p(\mathbf{q}'; \boldsymbol{\vartheta})} n_{q,h}(\mathbf{q}) \quad (3.52)$$

$$p_M(q | h; \boldsymbol{\vartheta}') = \frac{\max\{e(q, h) - d_M, 0\}}{\sum_{q'} e(q', h)} + \lambda(h) p_{M-1}(q | \bar{h}) \quad (3.53)$$

$$\hat{e}(q, \bar{h}) := \sum_{h \in \bar{h}} \min(e(q, h), d_M) \quad (3.54)$$

$$P_{M-1}(q | \bar{h}) = \frac{\hat{e}(q, \bar{h})}{\sum_{q'} \hat{e}(q', \bar{h})} \quad (3.55)$$

where h denotes the sequence of preceding joint units $h_j = (q_{j-M+1}, \dots, q_{j-1})$, and \bar{h} and the reduced history $\bar{h}_i = (q_{i-M+2}, \dots, q_{i-1})$. The above equations are iterated until convergence. The discount parameters d_1, \dots, d_M are optimized on a held-out set using Powell's method (Press *et al.* 1992).

Transcription

Having estimated the model, equation (1) can be applied to phonemically transcribe unseen words. In producing the phonemic transcription from the orthographic form, we usually restrict ourselves to approximating the sum in Equation (3.47) by the maximum.

$$p(\mathbf{g}, \boldsymbol{\varphi}) \approx \max_{q \in S(\mathbf{g}, \boldsymbol{\varphi})} p(\mathbf{q}) \quad (3.56)$$

This means, we look for the most likely grapheme sequence matching the given spelling and project it onto the phonemes.

$$\varphi(g) = \varphi \left(\underset{q \in Q^* | g(q)=g}{\operatorname{argmax}} p(q) \right) \quad (3.57)$$

N-best list generation is a theoretically pleasing way of generating alternative pronunciation candidates. We use the relation of alternative pronunciation candidates to compute the posterior probability of a translation:

$$p(\varphi | g) = \frac{\sum_{q \in S(\mathbf{g}, \boldsymbol{\varphi})} p(q)}{p(g)} \quad (3.58)$$

In practice we find that it is not necessary to carry out the summation in the numerator of Equation (3.58). Therefore, we will use the following approximation in our experiments:

$$p(\varphi|g) \approx \frac{\max_{q \in S(g, \varphi)} p(q)}{p(g)} \quad (3.59)$$

3.6.3.3 Experiments

Many Natural Language Processing applications benefit from a fixed defined vocabulary. On the one hand, the vocabulary size should be small for robust estimation of the model parameters, but on the other hand, it should be large to cover all task relevant words.

The vocabulary of a speech recognition system is defined by the language model vocabulary and the pronunciation lexicon. The transcription system can only recognize words which are modeled in both system dictionaries.

This chapter describes the application of the grapheme-to-phoneme converter to automatically generate pronunciations for 50,000 undiacritized Arabic words. If manually done, this task would have been very time consuming and would have required Arabic language expertise.

Text data processing

The processing of the data was performed by SRI. The text data sources were FBIS, TDT-4, GALE BN and BC transcripts (manual transcriptions and web data), Arabic GIGAWORD, CMU web data, CU web data, and UN data, for a total of about 1.1B words. All text data was processed using the MADA tool from Columbia University (Habash and Owen 2005), which uses a statistical morphological tagger to fix common errors or ambiguities in the Arabic orthography, like the presence of Hamzas above or below the letters Alef, Yeh and Waw, and the marking of final Yeh and Teh Marbuta. Other post-processing scripts were used to expand numbers and dates in the text data to words, remove non-lexical tags and punctuation marks, and normalize the orthography of some frequent names.

Vocabulary selection

vocab. size	64k	128k	256k	500k	600k
Eval06	5.36	3.09	2.1	1.2	1.07
Dev07	3.88	2.05	1.34	0.69	0.58

Table 3.53: Effect of vocabulary size on OOV rate [%].

Since Arabic is a morphologically rich language, the setup of the system vocabulary is a challenging task. The vocabulary size has to be fairly high compared to other languages to achieve a low out-of-vocabulary (OOV) rate. Table 3.53 shows the OOV

rates on the GALE Eval06 and Dev07 sets for different vocabulary sizes. SRI used the approach described by Venkataraman and Wang (2003) to choose the vocabulary.

Source lexicon generation

Diacritics are usually omitted in written Arabic texts. The diacritical marks are used for disambiguation and clarify the actual pronunciation. In the NIGHTINGALE system the diacritized MADA output was used as basis of the pronunciation lexicon. The mapping between the diacritized orthography and the phonemic representation is fairly straightforward. It is mostly a 1 - 1 mapping with a few rules that describe the pronunciation of Arabic. The pronunciations are constructed using a set of 34 speech phonemes.

Automatic lexicon augmentation

M-gram	1	2	3	4	5	6	7
PER [%]	39.1	28.6	23.7	18.7	16.3	15.35	15.44

Table 3.54: Grapheme-to-phoneme conversion results on held-out set.

MADA failed to produce output for about 50k words out of the 600k vocabulary. For these words we used the automatic pronunciation generation software to obtain their pronunciations. We trained the statistical grapheme-to-phoneme conversion model on a selected part of the source pronunciation lexicon. We set the length constraints L equal to 1 and trained up to a 7-gram model. Table 2 presents the grapheme-to-phoneme conversion results for different M-gram models on the heldout set. Accuracy of grapheme-to-phoneme conversion is measured in terms of phoneme error rate (PER), which is the edit distance between the automatic transcription result and reference pronunciation divided by the number of phonemes in the reference pronunciation.

The 6-gram model was used to generate the top five most likely pronunciations for words which pronunciations are not covered by the source lexicon. For the 2007 GALE evaluation the automatically augmented pronunciation lexicon was used by the SRI/NIGHTINGALE Arabic ASR system (Vergyri *et al.* 2008). The system obtained a WER of about 10% on the Eval07 broadcast news (BN) test set.

3.6.3.4 Conclusion

In developing a grapheme-to-phoneme conversion system for a new language it is inconvenient to write and update alignment rules for all words by hand.

We presented a fully data-driven, language in-dependent grapheme-to-phoneme converter and we described its application to generate multiple pronunciations for undiacritized words for the Arabic NIGHTINGALE system.

3.6.4. CMU-InterACT 2008 Arabic Speech Recognition System for GALE

Authors: Udhyakumar Nallasamy, Ian Lane, Mark Fuhs, Mohamed Noamany, Yik-Cheung Tam, Qin Jin and Tanja Schultz

3.6.4.1 Introduction

This section discusses the development and improvements in the CMU Modern Standard Arabic (MSA) transcription system and cross-site system combination efforts as part of Rosetta submission for the 2008 GALE evaluation.

Vowels are omitted in Written MSA, thus, native speakers add them during reading. Vowels are written only in children books or traditional religious books. We investigate predicting the vowels in the transcriptions using a morphological analyzer and Arabic Tree-Bank. Vowelized acoustic models give us significant improvement over graphemic, unvowelized models. We report the accuracy of the vowelized and unvowelized systems independently and cross-adapted systems. Models used in our system were trained using more than 1500 hours of transcribed audio data and over a billion words from 12 different news sources. We report our results on Dev 07 and Dev 08 datasets supplied by Linguistic Data Consortium (LDC).

3.6.4.2 System Overview

Our MSA speech recognition system is implemented using the Janus Recognition Toolkit (JRtk) (Finke *et al.* 1997), featuring IBIS decoder (Soltau *et al.* 2001). Automatic speaker clustering of the manually segmented input audio is performed before decoding. Segments are clustered into speaker-specific clusters using Bayesian Information Criterion (BIC) to enable speaker/cluster-specific adaptation and normalization. A detailed overview of the clustering step is presented by Hsaio *et al.* (Section 3.5.3).

We extract power spectral features using a FFT with a 10ms frame-shift and a 16ms Hamming window from the incoming 16kHz audio signal. We use them to compute 13 Mel-Frequency Cepstral Coefficients (MFCC) per frame and perform a cepstral mean and variance normalization on a cluster basis. To incorporate dynamic features we concatenate 15 adjacent MFCC frames (± 7) and project 195 dimensional features into a 42 dimensional space using a Linear Discriminate Analysis (LDA) transform. We use the context-dependent codebooks as classes for the LDA transform. In addition to LDA, we apply a single maximum likelihood trained Semi-Tied-Covariance (STC) matrix (Gales 1999).

Our STT transcription engine employs a two pass decoding strategy. In the first pass, the decoder uses speaker-independent acoustic models without Vocal Tract Length Normalization (VTLN) and speaker adaptation. The hypotheses of a cluster from the first pass are then used to estimate the warping factors to warp the power spectrum using the maximum likelihood approach (Zhan and Westphal 1997). After computing the VTLN warping factors, the same hypothesis are used to estimate Feature Space Adaptation (FSA) transform using Constrained Maximum Likelihood Linear Regression (CMLLR).

Model space adaptation is then performed using MLLR with multiple regression classes (Gales 1998). Regression classes are obtained by clustering the Gaussians in the acoustic model. The second pass decoding uses a speaker adaptive acoustic model with feature and model space adaption per speaker.

For the development of context dependent acoustic models we applied an entropy-based polyphone decision tree clustering process using context questions of maximum width ± 2 , resulting in shared quinphones. In addition, we included word-boundary tags into the pronunciation dictionary, which can be used as questions in the decision tree. Both the vowelized and the unvowelized systems use 6000 phonetically-tied quinphones with a maximum of 150 gaussians per senone.

3.6.4.3 Language Modeling

We used several different sources to train our language model. The Arabic Gigaword corpus distributed by LDC is the major text resource for language modeling. As this corpus only covers broadcast news, we harvested Arabic talk show transcripts from Al-Jazeera (www.al-jazeera.net) and BN transcripts from Al-Akhbar and Akhbar Elyom. Details of this data collection are explained by Noamany *et al.* (2007). Acoustic transcripts from FBIS, TDT-4, GALE BN and BC were also used. The total number of words in the corpus amounted to 1.1 billion.

To improve coverage and specificity for both BN and BC data, we trained 11 different 4-gram language models and interpolated them to create the final LM. For language model training the SRILM toolkit (Stolcke 2002) was used. Interpolation weights were selected based on a heldout data set selected from BN and BC sources. The final LM contained 692M n-grams and a vocabulary of 737K words.

3.6.4.4 Acoustic Modeling

We trained two different acoustic models which are the basic blocks in a multi-pass decoding strategy: Unvowelized models which uses one-to-one letter to sound mapping without any short vowels and vowelized models in which short vowels are predicted for words in the dictionary and explicit models are trained for these acoustic units.

Unvowelized Acoustic Models

System	Dev07	Dev 08
SI	20.4%	22.8%
SA	17.0%	19.0%

Table 3.55: WER of Unvowelized models on Dev 07 and Dev 08.

Arabic is a phonetic language with a close correspondence between its letters and sounds. We used grapheme-to-phoneme rules similar to (Billa *et al.* 2002) to build the pronunciation dictionary. Our unvowelized system contains 37 phones with three special phones for silence, non-speech events and non-verbal effects such as hesitation. We preprocessed the text by mapping the three shapes of the grapheme for glottal stops to one shape at the beginning of the word since these are frequently mis-transcribed. This

preprocessing step gave us improvements in perplexity and the final WER in our previous experiments (Noamany *et al.* 2007). We used a vocabulary of 737K words selected by including all words appearing in the acoustic transcripts and the most frequent words occurring in the LM. Table 3.55 shows the Word Error Rate (WER) of our Speaker-Independent (SI) and Speaker-Adaptive (SA) Unvowelized system on the Dev 07 and Dev 08 test set.

Vowelized Acoustic Models

Arabic is a morphological language in which short vowels are omitted in the written form. These vowels carry grammatical case information and influence the meaning of a word. To restore vowels for the 737K vocabulary, we performed the following steps:

- Buckwalter morphological analyzer (BMA)
- If a word is not vowelized by the analyzer, we check for its vowelization in the LDC Arabic Tree-Bank
- If the word did not appear in any of those, we used the same letter-to-sound rules used in the unvowelized word form

To reduce the number of pronunciation variants we performed a forced alignment and excluded pronunciations that did not occur in the training corpus. The final vowelization step resulted in 2.3M pronunciations for the 737k words in our vocabulary (an average 3.1 pronunciations per word). The vowelized system uses the same vocabulary and training procedure as used in the unvowelized case. Pronunciation weights were computed based on frequency of occurrence during forced alignment or the Arabic Tree-Bank. Incorporating pronunciation weights reduced WER by 0.9% absolute on Dev 07 compared to decoding without them (Table 3.56).

System	Dev07
Vowelized SA	17.1%
Vowelized SA +Pronunciation weights	16.2%

Table 3.56: Effect of pronunciation weights on WER.

We investigated the use of modified topology for short vowels following (Soltau *et al.* 2007). We found that modeling short vowels using a three state model with additional skip transitions improved our results by a further 0.7% absolute on Dev 07 test set, as shown in Table 3.57.

Henceforth, all of our vowelized models use three state skip models for short vowels and pronunciation weights during recognition. Table 3.58 shows the performance of the final vowelized models in terms of Word Error Rate (WER) on Dev 07 and Dev 08.

System	Dev07
Vowelized SA + Pronunciation Weights + 3-state short vowels	16.2%
Vowelized SA + Pronunciation weights + 3-state short vowels with skip	15.5%

Table 3.57: Effect of pronunciation weights on WER.

System	Dev07	Dev 08
SI	18.7%	20.4%
SA	15.5%	16.7%

Table 3.58: WER of Vowelized models on Dev 07 and Dev 08.

3.6.4.5 Cross-Adaptation and LSA Rescoring

We performed cross-adaptation based on three pass decoding with several different combinations of our Vowelized (V) and Unvowelized (U) acoustic models. The WER of these cross-adapted models are summarized in Table 3.59.

System	Dev 07	Dev 08
V-SI x V-SA x U-SA	14.7%	16.9%
V-SI x U-SA x V-SA	14.0%	15.8%
U-SI x U-SA x V-SA	14.3%	15.9%

Table 3.59: WER of 3-pass cross-adapted models on Dev 07 and Dev 08.

The lattices of the best system (U-SI x U-SA x V-SA) were then rescored using Bigram based correlated Latent Semantic Analysis (LSA), which reduced the WER by 0.5% absolute on Dev 07 and Dev 08. Detailed information on LSA is explained by Tam *et al.* (Section 3.4.3).

System	Dev 07
2007 models (U-SI x U-SA x V-SA) + LSA Rescoring	16.3%
2008 models (U-SI x U-SA x V-SA) + Bigram LSA rescoring	13.8%

Table 3.60: Improvement of 2008 best system over 2007 single best system.

Overall, our 2008 GALE single best ASR reduced the WER by 15% relative (from 16.3% to 13.8%) on Dev 07 compared to last year's single best system. The reduction in WER stems from using additional training data (~ 1.0-1.3% absolute gain), pronunciation weights in the dictionary (~ 0.3-0.6% absolute gain), modified HMM topology (three state models with skip transitions) for short vowels (~ 0.3-0.6% absolute gain) and improved bigram LSA based lattice rescoring (~ 0.4-0.5% absolute gain). Table 3.60 illustrates the improvement.

System	Dev 07	Dev 08
IBM x V-SA + Bigram LSA rescoring	11.8%	13.1%
IBM x U-SA + Bigram LSA rescoring	12.8%	14.2%

Table 3.61: WER of IBMxCMU cross-adapted models on Dev 07 and Dev 08.

We also employed cross-site adaptation by adapting CMU’s vowelized and unvowelized models on IBM’s best system hypothesis followed by Bigram LSA rescoring. The results are listed in Table 3.61.

3.6.4.6 System Selection and Multi-System Combination

In the GALE-2008 evaluation eight STT (speech-to-text) transcription engines were implemented within the Rosetta Team. Individual systems differed by the phonetic units used for acoustic modeling (vowelized, unvowelized, and different modeling of short vowels), if cross-adaptation between systems had been applied, and the method of language model rescoring used. On Dev 07 individual systems obtained WERs ranging from 9.1% to 11.8%. Due to the large number of STT engines, the iterative system selection approach described by Hsaio *et al.* (Section 3.5.3) was applied to select the optimal n -way combination-set. These systems were then combined using lattice combination to produce the final STT output. During the selection process Dev 07 was used as the tuning set, to optimize combination weights, and Dev 08 was used for system selection. The WERs of the top performing systems and the best n -way system combinations are shown in Table 3.62.

System	Word Error Rate in % (Sentence Error Rate in %)	
	Dev 07 (tuning)	Eval 07 (test)
1	9.3 (64.1)	10.6 (64.5)
2	9.9 (66.4)	11.4 (66.4)
3	9.9 (65.3)	11.0 (65.4)
1+2	9.1(64.3)	10.3 (64.1)
1+2+3	9.0 (59.9)	10.6 (57.1)

Table 3.62: Performance of Multi-system lattice combination. 1. IBM: Universal Background Model (UBM) cross-adapted on Vowelized ASR; 2. IBM: Unvowelized Broadcast News ASR; 3. IBM: Neural Network rescored lattices.

The best two way system combination “1+2” obtained an absolute reduction in WER of 0.2 points on Dev 07 and 0.3 points on Dev 08, compared to the single best system “1”. A relative reduction in WER of 3% was obtained on unseen data. All three way combinations obtained higher WERs on unseen data, thus the pair-wise system was used as the final STT output. In addition to improving transcription quality, lattice combination was also found to improve translation accuracy. When the IBM phrase-based SMT system was applied to the combined system output, Translation Error Rate (TER) was reduced by 0.5 points (from 51.90% to 51.40%) and BLEU improved by 0.16 points (from 32.69 to 32.85), compared to the single best system “1” (Dev 07).

3.6.4.7 Conclusion

We presented CMU efforts in the 2008 GALE Arabic STT evaluation within the Rosetta team. We achieved an overall relative reduction in WER of 15% for our single best system compared to the previous year using techniques like better pronunciation modeling and improved Bigram LSA rescoring. Future work includes further improvements using discriminative training using MMI and MPE techniques (Povey 2003) and better coupling between ASR and MT systems.

3.6.5 Arabic Speech Transcription at IBM

Authors: Hagen Soltau, George Saon, Brian Kingsbury, Hong-Kwang Kuo, Lidia Mangu, and Daniel Povey

3.6.5.1 Introduction

The goal of DARPA's Global Autonomous Language Exploitation (GALE) program is to make foreign language (Arabic and Chinese) speech and text accessible to English monoglots, particularly in military settings. We focus in this section on Arabic broadcast transcription, although many of the techniques we describe have been successfully applied to our Chinese and English broadcast transcription systems.

3.6.5.2 System Overview

The IBM system for Arabic Broadcast News consists of unvowelized and vowelized acoustic models, cross-adapted to each other. The unvowelized system is strictly graphemic, while the vowelized system models diacritics and short vowels.

3.6.5.2.1. Acoustic Training Data

We used the following corpora for acoustic model training in various experiments presented here:

- 85 hours of FBIS and TDT-4 audio with transcripts provided by BBN,
- 51 hours of transcribed GALE data provided by the LDC for the GALE Phase 1 (P1) evaluation,
- 700 hours of transcribed GALE data provided by LDC for the Phase 2,
- 5.6 hours of BN data with manually vowelized transcripts provided by BBN (BNAT-05),
- 5.1 hours of BN data with manually vowelized transcripts provided by BBN (BNAD-05),
- 500 hours of transcribed Iraqi data (TRANSTAC),
- 1800 hours of untranscribed audio from the EARS BN-03 corpus, and
- 10000 hours of untranscribed audio collected at IBM Research (TALES).

Source	Hours	Source	Hours
Al Arabiya	2455	LBC (Lebanese) Flash News	88
Al Jazeera Morning News	306	LBC International News	161
Al Jazeera Midday News	326	Voice of America	184
Al Jazeera News Bulletin	975	Al Alam	47
Dubai News	27	NBN News	131
ESC News	81		

Table 3.63: Source variability in Arabic broadcast training data.

BNAT-05 and BNAD-05 were used only for a comparison of flat start training to manual data for the initialization of vowelized models. The TRANSTAC data was used only for experiments on dialect modeling. The TALES data was used only for experiments on large scale unsupervised training, including tests of dialect models. The evaluation system described in Section 3.6.5.2.2 was trained only on data available to all GALE program participants: FBIS, TDT-4, GALE P1 and P2, and EARS BN-03. Arabic broadcast news and conversations are highly variable, coming from many different broadcasters and containing a mixture of Modern Standard Arabic (MSA) and dialectical Arabic. To illustrate this variability, Table 3.63 provides a breakdown of a sample comprising 130 hours of GALE transcribed data, 750 hours of untranscribed EARS BN-03 data, and 5600 hours of untranscribed TALES data by source. Note that only the predominant sources in the sample are listed; there is also a long tail containing many additional sources, which are represented by smaller amounts of audio.

3.6.5.2.2. Eval07 System Description

The operation of our system follows the steps in Figure 3.33: **(1)** segmentation of the audio into speech and non-speech segments, **(2)** clustering of the speech segments into speaker clusters, **(3)** speaker independent (SI) decoding of the speech segments, **(4)** estimation of VTLN warp factors based on the SI output, **(5a)** estimation of FMLLR and MLLR for the unvowelized acoustic models based on the SI output, **(5b)** estimation of FMLLR and MLLR for the vowelized acoustic models based on the SI output, **(6a)** speaker-adapted decoding with unvowelized acoustic models (U-SA), **(6b)** speaker-adapted decoding with vowelized acoustic models (V-SA), **(7a)** estimation of FMLLR and MLLR for U-SA based on the V-SA output, **(7b)** estimation of FMLLR and MLLR for V-SA based on the U-SA output, **(8a)** re-decoding with U-SA, **(8b)** re-decoding with V-SA, **(9a)** rescoreing of the lattices from (8a) with a neural-network language model, **(9b)** rescoreing of the lattices from (8b) with a neural-network language model, and **(10)** lattice combination of the lattices from (9a) and (9b). The performance of various decoding steps on DEV-07 and EVAL-07-P2 is summarized in Table 3.64.

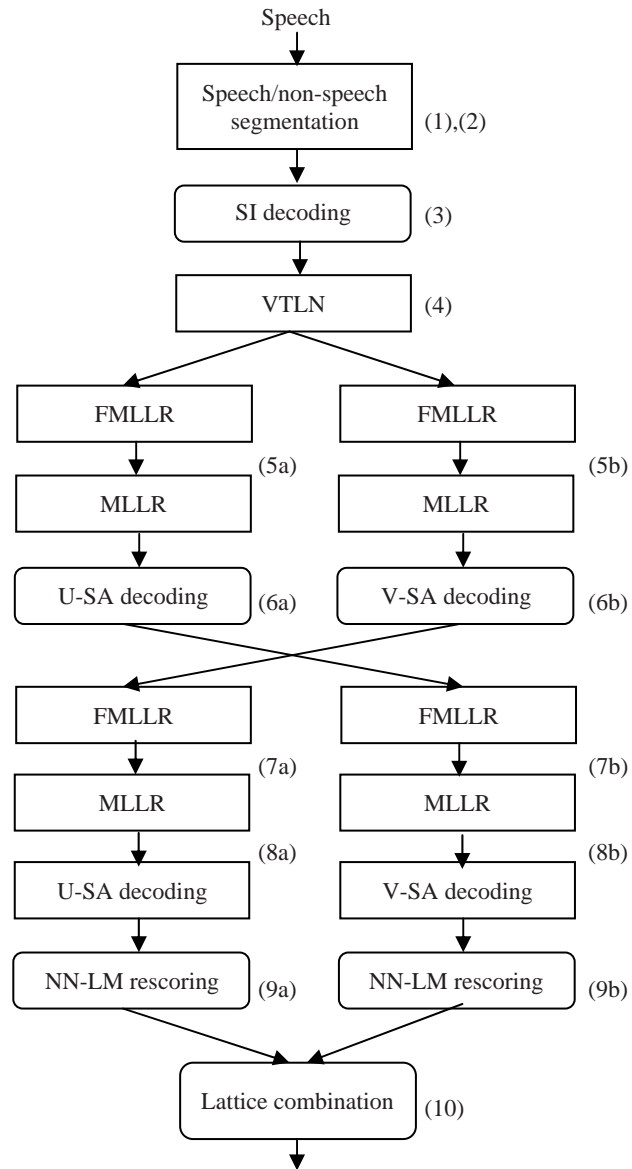


Figure 3.33: System diagram. Oval boxes represent decoding or rescoring steps.

Step	Decoding pass	DEV-07	EVAL-07-P2
(3)	U-SI	24.5%	22.4%
(6a)	U-SA	12.2%	10.1%
(6b)	V-SA	11.4%	9.4%
(8a)	U-SA	11.1%	8.7%
(8b)	V-SA	10.7%	8.6%
(9a)	LM resc. (8a)	10.4%	8.5%
(9b)	LM resc. (8b)	10.2%	8.0%
(10)	Lattice comb.	9.9%	8.0%

Table 3.64: Word error rates of different decoding steps on DEV-07 and EVAL-07-P2.

3.6.5.2.3. Acoustic Modeling

The input speech is represented by VTL-warped PLP features and a context window of eight frames. The PLP features are mean and variance normalized on a per speaker basis. An LDA transform is used to reduce the feature dimensionality to 40. The maximum-likelihood training of the acoustic model is interleaved with an estimation of a global semi-tied full covariance (STC) (Gales 1999).

System	Nb. leaves	Nb. Gaussians
SI	3.5K	150K
U-SA	5.0K	400K
V-SA	4.0K	400K

Table 3.65: GALE P2.5 evaluation system statistics.

Words in the recognition lexicon are represented as sequences of phones, and phones are modeled with three state left-to-right HMMs that do not permit state skipping. All models have pentaphone crossword acoustic context. The numbers of context-dependent states and Gaussian mixture components used in our GALE P2 evaluation system are summarized in Table 3.65.

The speaker-adapted models are discriminatively trained using a variety of techniques operating in both feature and model space. One such technique is a feature-space transformation called feature-space minimum phone error (fMPE) (Povey *et al.* 2005). For model-space training, we experimented with minimum phone error (MPE) and boosted maximum mutual information (BMMI) (Povey *et al.* 2008), a new form of discriminative training based on large margin classification. Additionally, we experimented with separate training of the transform and the models in two stages. In the first stage, we estimate the fMPE transform using the original set of ML-trained models. In the second stage, we rebuild the decision trees and new ML models in the resulting fMPE space, finishing with model-space discriminative training of the new ML models.

Training style	Features	Models	WER
-	none	ML	17.1%
Integrated	fMPE	ML	14.3%
Integrated	fMPE	MPE	13.0%
Integrated	fBMMI	ML	14.4%
Integrated	fBMMI	BMMI	13.2%
Two-stage	fMPE	ML	14.3%
Two-stage	fMPE	BMMI	12.4%

Table 3.66: Discriminative training results for the unvoiced speaker-adapted models on DEV-07.

In Table 3.66, we show various discriminative training results on 1400 hours of data, comparing MPE and BMMI, and single-stage versus two-stage training for unvoiced speaker-adapted models on DEV-07. It appears that fMPE and fBMMI have similar performance (14.3% versus 14.4%) and that two-stage training with alternating criteria (MPE for feature-space and BMMI for model-space) gives the best results. More

experiments are needed to determine whether this is due to the two-stage training or to the different criteria.

3.6.5.2.4. Recognition Vocabulary

One challenge in Arabic speech recognition is that the rich morphological structure of Arabic induces a large number of word forms. We dealt with this challenge by simply increasing the size of our recognition vocabulary. The selection criterion is word frequency with source dependent cutoff values (placing emphasis on acoustic transcripts and MT vocabulary). Table 3.67 shows out-of-vocabulary (OOV) rates and word error rates on DEV-07 for an ML-trained vowelized system (without pronunciation probabilities) as a function of vocabulary size. The benefit of increasing the vocabulary from 129K to 737K words is 1.2% absolute (10.7% relative). While these vocabularies are quite large, the search is still manageable. The memory footprint for a static FSM decoder (Saon *et al.* 2003) is about 2.5GB with a 3M 4-gram LM and 2.5M pronunciations, while the memory footprint for a dynamic network decoder similar to (Soltau *et al.* 2001) is 1.6GB with a 490M 4-gram LM (memory mapped) and 2.5M pronunciations. With open beams on a Linux server with a 3.6GHz CPU, the static decoder runs 2.5 times slower than real time (2.5xRT), while the dynamic decoder runs at 2.9xRT.

Nb. words	Nb. pron.	OOV Rate	WER
129K	599K	3.0%	11.1%
343K	1372K	1.4%	10.2%
589K	2173K	0.9%	10.0%
737K	2563K	0.8%	9.9%

Table 3.67: OOV and word error rates on DEV-07 as a function of vocabulary size.

3.6.5.3 Unsupervised Training

We ran a number of experiments on the use of unsupervised training. Our approach is standard (Kemp and Waibel 1998; Zavaliagkos and Colthurst 1998): we decode the data using an unvowelized speaker adapted system (left side of Figure 3.33), then select a subset of the data based on per utterance average word posteriors, computing the word posteriors from confusion networks.

Posterior thresh.	Hours	WER
1.0 (baseline)	0	17.2%
0.95	751	15.9%
0.9	1482	15.7%

Table 3.68: Amount of unsupervised data and word error rates on RT-04 as a function of posterior threshold when using 135 hours of supervised data.

In our first experiments (Soltau *et al.* 2007), early in the GALE Program, we used 1800 hours of untranscribed audio from the EARS BN-03 corpus and 135 hours of GALE

Phase 1 supervised data (cf. Section 3.6.5.2.1). Varying amounts of unsupervised data were selected by varying the threshold on the average word posterior. As can be seen in Table 3.68, satisfactory performance is obtained with 751 hours of unsupervised data.

Total	Unsup.	Gaussians	EVAL-07
848	0	400K	17.7%
2330	1482	400K	18.3%
2330	1482	800K	18.0%
7917	7069	400K	18.5%
7917	7069	800K	18.3%

Table 3.69: Effect of unsupervised training data and number of Gaussians using 848 hours of supervised data.

Over the course of the GALE Program, more supervised data was produced by the LDC. Once over 800 hours of supervised training data were available, we again checked to see if the additional unsupervised data were still useful. We compared the performance of models trained on supervised data only (848 hours) to models trained on additional unsupervised data of 1482 hours. While adding 1482 hours of unsupervised data helped by 1.5% when only 135 hours of supervised data was available (Table 3.68), we now see an increase in error rate between 0.6% (Table 3.69).

Later, we conducted a third experiment on unsupervised training in which we used a significantly larger amount of data: an internal collection of 10,000 hours of untranscribed audio from Arabic television channels (TALES data collection). In addition to selecting data based on average word posteriors, we also used language identification (Navratil 2006) to exclude segments in foreign languages (mostly English). About 5600 hours of audio remained after selection, which was added to the 848 hours of supervised GALE data and the 1482 hours of unsupervised EARS BN-03 data. The results in Table 3.69 show that the improvement from unsupervised training data disappears completely when more supervised training data is available, and that this effect appears to be independent of the size of the acoustic model.

3.6.5.4 Vowelization

One challenge in Arabic speech recognition is that there is a systematic mismatch between written and spoken Arabic. Short vowels and diacritics are normally omitted in written Arabic. An excellent discussion of the Arabic language and automatic speech recognition can be found in Kirchhoff *et al.* (2003).

There are two approaches to handling this mismatch between the acoustics and transcripts. In the “unvowelized” approach, words are modeled graphemically, in terms of their letter sequences, and the acoustics corresponding to the unwritten diacritics are implicitly modeled by the Gaussian mixtures in the acoustic model. In the “vowelized” approach, words are modeled phonemically, in terms of their sound sequences, and the correct vocalization of transcribed words is inferred during training. Note that even when vowelized models are used the word error rate calculation is based on unvowelized references. The vowelized forms are mapped back to unvowelized forms in scoring (NIST style scoring).

One of the biggest challenges in building vowelized models is initialization: how to obtain a first set of vowelized models when only unvowelized transcripts are available. One approach is to have experts in Arabic manually vowelize a small training set (Messaoudi *et al.* 2006). The obvious disadvantage is that this process is quite labor intensive, which motivates researchers to explore automated methods (Afify *et al.* 2005). Following the recipe of Afify *et al.* (2005), we discuss our bootstrap procedure and some issues related to scaling up to large vocabularies.

We use the Buckwalter Morphological Analyzer (Version 2.0) (Buckwalter 2004), and the Arabic Treebank to generate vowelized variants of each word. The pronunciation of each variant is modeled as the sequence of letters in the diacriticized word, including the short vowels. For *shadda* (consonant doubling), an additional consonant is added (same phone symbol), and for *sukun* (no vowel), nothing is added. We use a phone set of 47 phones.

The vowelized training dictionary has 243368 vowelized pronunciations, covering a word list of 64496 words. The vowelization rate is about 95%. For the remaining 5% of words that are not covered, we back off to unvowelized forms.

3.6.5.4.1. Flat-Start Training vs. Manual Transcripts

Our flat-start training procedure initializes context independent HMMs by distributing the data equally across the HMM state sequence. We start with one Gaussian per state, and increase the number of parameters using mixture splitting interleaved within 30 forward/backward iterations. Now, the problem is that we have 3.8 vowelized pronunciations per word on average, but distributing the data requires a linear state graph for the initialization step. To overcome this problem, in the first iteration of training we randomly select pronunciation variants. All subsequent training iterations operate on the full state graph representing all possible pronunciations.

We compare this approach to manually vowelized transcripts where the correct pronunciation variant is given. BBN distributed ten hours of manually vowelized development data (BNAD-05, BNAT-05) that we used to bootstrap vowelized models. These models are then used to compute alignments for the standard training set (FBIS + TDT4). A new system is then built using fixed alignment training, followed by a few forward/backward iterations to refine the models. The error rates in Table 3.70 suggest that manually vowelized transcripts are not necessary. The fully automatic procedure is only 0.2% worse. We opted for the fully automatic procedure in all our experiments, including the evaluation system.

Training Method	WER (RT-04)
Flat-start	23.0%
Bootstrap	22.8%

Table 3.70: Comparison of different Initialization methods for vowelized models.

3.6.5.4.2. Short Models for Short Vowels

Topology	RT-04	BCAD-05
3-state	19.0%	28.9%
2-state	18.5%	27.4%

Table 3.71: Comparison of different HMM topologies for short vowels.

We noticed that the vowelized system performed poorly on broadcast conversational speech. It appeared that the speaking rate is much faster, and that the vowelized state graph is too large to be traversed with the available speech frames. Recall that the acoustic models do not permit state skipping (Section 3.6.5.2.3). One solution is to model the three short vowels with a shorter, two state HMM topology. The results are shown in Table 3.71. The improvements on RT-04 (broadcast news) are relatively small; however, there is a 1.5% absolute improvement on BCAD-05 (broadcast conversations).

3.6.5.4.3. Vowelization Coverage of the Test Vocabulary

As mentioned before, we back off to unvowelized forms for those words not covered by Buckwalter and Arabic Treebank. The coverage for the training dictionary is pretty high at 95%. On the other hand, for a test vocabulary of 589K words, the vowelization rate is only 72.6%. The question is whether it is necessary to manually vowelize the missing words, or whether we can get around that by backing off to the unvowelized pronunciations. One way to test this – without actually providing vowelized forms for the missing words – is to look at the OOV/WER ratio. The assumption is that the ratio is the same for a vowelized and an unvowelized system if the dictionary of the vowelized system does not pose any problems. More precisely, if we increase the vocabulary and we get the same error reduction for the vowelized system, then, most likely, there is no fundamental problem with the vowelized pronunciation dictionary.

Vocab.	unvowelized		Vowelized			
	OOV	WER	Variants	Vowel. Rate	OOV	WER
129K	2.9%	20.3%	538K	90.4%	2.9%	19.8%
589K	0.8%	19.0%	1967K	72.6%	0.8%	18.3%

Table 3.72: OOV/WER ratio for an unvowelized and a vowelized system on RT-04.

For the unvowelized system, when increasing the vocabulary from 129K to 589K, we reduce the OOV rate from 2.9% to 0.8%, and we reduce the error rate by 1.3% (Table 3.72 columns 2 and 3). For the vowelized system, we see a similar error reduction of 1.5% for the same vocabulary increase (Table 3.72 columns 6 and 7). The system has almost 2 million vowelized pronunciations for a vocabulary of 589K words. The vowelization rate is about 72.6%. In other words, 27.4% of our list of 589K words are unvowelized in our dictionary. Under the assumption that we can expect the same OOV/WER ratio for both the vowelized and unvowelized system, the results in Table 3.72 suggest that the backoff strategy to the unvowelized forms is valid for our vowelized system.

3.6.5.4.4. Pronunciation Probabilities

Our decoding dictionary has about 3.3 pronunciations per word on average. Therefore, estimating pronunciation probabilities is essential to improve discrimination between the vowelized forms. We estimated the pronunciation probabilities by counting the variants in the 2330 hour training set.

System	RT-04	BCAD-05
Unvowelized	17.0%	25.4%
Vowelized	16.0%	26.0%
+ Pron. Prob	14.9%	25.1%

Table 3.73: Effect of pronunciation probabilities on WER.

The setup consists of ML models, and includes all the adaptation steps (VTLN, FMLLR, MLLR). The test sets are RT-04 (Broadcast News) and BCAD-05 (Broadcast Conversations). Adding pronunciation probabilities gives consistent improvements between 0.9% and 1.1% on all test sets (Table 3.73). Also, pronunciation probabilities are crucial for vowelized models; they almost double the error reduction from vowelization. We investigated several smoothing techniques and longer word context, but did not see any further improvements over simple unigram pronunciation probabilities.

3.6.5.4.5. Vowelization, Adaptation, and Discriminative Training

In this section, we summarize additional experiments with vowelized models. One interesting observation is that the gain from vowelization decreases significantly when more refined adaptation and training techniques are used. Table 3.74 shows the error rate of unvowelized and vowelized models with and without adaptation. The relative improvement from vowelization is more than 15% at the speaker independent level. However, after applying several normalization and adaptation techniques, the gain drops to 7.7% at the MLLR level.

Decoding pass	Unvowelized	Vowelized
SI	29.5%	25.0%
VTLN	26.2%	23.1%
FMLLR	23.0%	21.1%
MLLR	22.1%	20.4%

Table 3.74: Comparison of vowelized and unvowelized models at various adaptation passes, no pronunciation probabilities. RT-04 test set

Models	Unvowelized	Vowelized
ML	17.1%	14.8%
fMPE+MPE	12.2%	11.4%

Table 3.75: Comparison of vowelized and unvowelized models before and after discriminative training. Vowelized system uses pronunciation probabilities and 2-state HMMs for short vowels. Both systems use speaker adaptation: VTLN, FMLLR, and MLLR. DEV-07 test set

An even more drastic reduction of the vowelization gain is observed after discriminative training (Table 3.75). The vowelized setup includes the two state HMMs for short vowels and pronunciation probabilities. While discriminative training reduces the error rate by 4.9% for the unvowelized setup, we observed an error reduction of only 3.4% for the vowelized models.

It seems that standard adaptation and discriminative techniques can at least partially compensate for the invalid model assumption of ignoring short vowels, and the improvements from vowelization are subsequently reduced when using better adaptation and training techniques.

3.6.5.5 Modeling of Arabic Dialects in Decision Trees

As shown in Table 3.63, the training data comes from a variety of sources, and there are systematic variations in the data, including dialects (Modern Standard Arabic, Egyptian Arabic, Gulf Arabic, etc.), broadcasters (Al Jazeera, Al Arabiya, LBC, etc.), and programs (Al Jazeera morning news, Al Jazeera news bulletin, etc.).

The problem we face is how to build acoustic models on diverse training data. Simply adding more training data from a variety of sources does not always improve system performance. To demonstrate this, we built two acoustic models with identical configurations. In one case, the model was trained on GALE data (2330 hours, including unsupervised BN-03), while in the second case we added 500 hours of TRANSTAC data to the training set. TRANSTAC data contains Iraqi Arabic (Nadji spoken Arabic and Mesopotamian Arabic). Both GALE and TRANSTAC data are Arabic data, but they represent very different dialects and styles. The model trained on additional TRANSTAC data is 0.4% worse than our GALE model (see Table 3.76).

Acoustic model	WER (DEV-07)
2330h GALE	19.2%
+ 500h TRANSTAC	19.6%

Table 3.76: Acoustic models trained without and with additional TRANSTAC data.

Both acoustic models used 400K Gaussians, a comparatively large number that should be able to capture the variability of different data sources. Furthermore, we did not find that increasing the number of Gaussians could offset the reduced performance caused by the addition of unmatched training data in previous experiments with large amounts of diverse unsupervised training data (see Table 3.69).

Adding more training data will not always improve performance, we need to find which part of the training data is relevant. Training separate models for each category or dialect requires sufficient training data for each category, significantly more human and computational effort, and algorithms that reliably cluster training and test data into meaningful categories. We propose instead to model dialects directly in the decision trees: a data driven method for building dialect-specific models without splitting the data into different categories.

Similar approaches to use additional information in decision trees can be found in the literature. (Reichl and Chou 1999) used gender information to specialize decision

trees for acoustic modeling. Speaking rate, SNR, and dialects were used by Fugen and Rogina (2000), and questions about the speaking mode were used by Soltau and Waibel (2000) to build models for hyper-articulated speech.

3.6.5.5.1. Decision Trees with Dialect Questions

We extend our regular decision tree algorithm by including non-phonetic questions about dialects. The question set contains our normal phonetic context questions, as well as, dynamic questions regarding dialect. Dialect questions compete with phonetic context questions to split nodes in the tree. If dialect information is irrelevant for some phones, it will simply be pruned away during tree training.

The training of dialect specific trees and Gaussian mixture models is straightforward. The decision tree is grown in a top-down clustering procedure. At each node, all valid questions are evaluated, and the question with the best increase in likelihood is selected. We use single Gaussians with diagonal co-variances as node models. Statistics for each unclustered context-dependent HMM state are generated in one pass over the training data prior to the tree growing. Dialect information is added during HMM construction by tagging phones with additional information. The additional storage cost this entails is quite significant: the number of unique, unclustered context-dependent HMM states increases roughly in proportion to the number of different tags used.

The questions used for tree clustering are written as conjugate normal forms. Literals are basic questions about the phone class or tag for a given position. This allows for more complex questions such as: *Is the left context a vowel and is the channel Al Jazeera (-1=Vowel && 0=AlJazeera)*. Similarly, more complex questions on the source may be composed. For example, to ask for Al Jazeera, one would write *(0=AlJazeeraMorning or 0=AlJazeeraAfternoon)*. The idea is to allow a broad range of possible questions, and to let the clustering procedure select the relevant questions based on the training data. The questions cover all the channel and dialect information available from the audio filenames.

We used this technique to build a dialect-dependent decision tree for the acoustic models trained on the combined GALE and TRANSTAC data. We generated a tree with 15865 nodes, and 8000 HMM states. Approximately 44% of the states depend on the dialect tag, so the remaining 56% of the models are shared between two very different data sources: GALE and TRANSTAC.

3.6.5.5.2. Building Static Decoding Graphs for Dynamic Trees

Training dialect-specific models is relatively easy; however, decoding with such models is more complicated if a statically compiled decoding graph is used. The problem is that the decision tree contains dynamic questions that can be answered only at runtime, and not when the graph is compiled. The solution for this problem is to separate the decision tree into two parts: a static part containing only phonetic questions, and a dynamic part for the dialect questions. The decision tree is reordered such that no dynamic question occurs above a static question. The static part of the decision tree can be compiled into a decoding graph as usual, while the dynamic part of the tree is replaced

by a set of virtual leaves. The decoder maintains a lookup table that transforms each virtual leaf to a corresponding dialect-specific leaf at run-time.

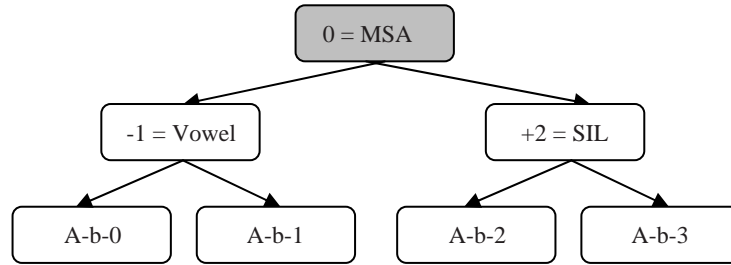


Figure 3.34: Decision Tree with dialect questions. Each question is of the format *ContextPosition = Class*. Leaves are marked with HMM states *A-b-0,..* (phone /A/, begin state).

In the following, we explain how to reorder the decision tree such that dynamic questions do not occur above static questions. Figure 3.34, Figure 3.35, and Figure 3.36 illustrate the process. In this example, the root of the tree is marked with the question *0=MSA*. If the center phone is marked as Modern Standard Arabic (MSA), the right branch will be chosen, otherwise the left branch.

The reordering algorithm consists of a sequence of transformation steps. In each step, a dynamic question node is pushed down one level by replacing it with one of its children nodes. Figure 3.35 shows the tree after applying one transformation step. In each transformation step, the selected dynamic question node is duplicated together with one of its branches. In this example, the right branch starting with *+2=SIL* is duplicated. The node to be moved up (promoted) (*-1=Vowel* in Figure 3) is separated from its sub-trees and moved up one level, with the duplicated sub-trees becoming its children. The sub-trees that were originally children of the promoted node become its grandchildren. The resulting tree is equivalent to the original tree, but the dynamic question is now one level deeper.

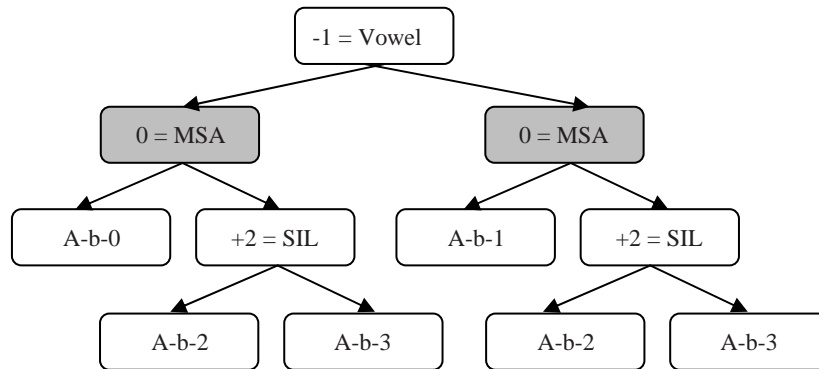


Figure 3.35: Decision Tree after one transformation step.

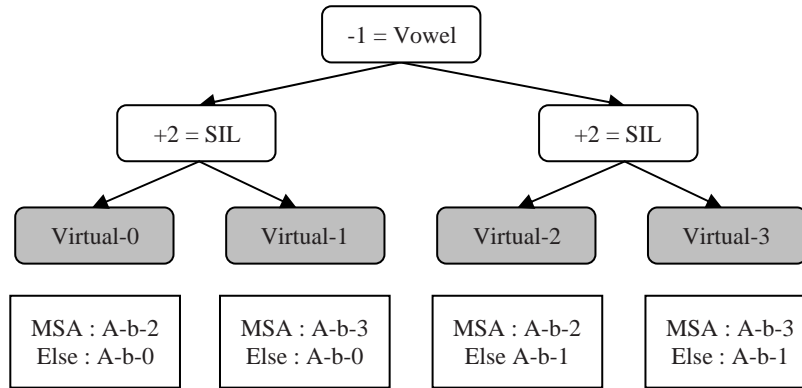


Figure 3.36: Reordered Decision Tree. Dynamic questions are replaced by virtual leaves. The original root $0=MSA$ was replaced by the left child $-1=Vowel$, and a new copy of $0=MSA$ was created.

The reordering procedure terminates when no transformation step can be applied. In the worst case, this procedure causes the decision tree's size to grow exponentially; however, in practice we observe only moderate growth. Table 3.77 summarizes the tree statistics after reordering. The number of internal nodes grew by only a factor of two, which is easily managed for decoding graph construction. The last step is to separate the static and dynamic tree parts. The dynamic question nodes are replaced by virtual leaves for the graph construction. The virtual leaves correspond to lookup tables that map virtual leaves to physical HMM states at run time. The static decoding graph can now be constructed using the tree with virtual leaves. At run time, dialect information is available, and virtual leaves can be mapped to the corresponding physical HMM states for acoustic score computation.¹²

Number of nodes	15865
Number of leaves	8000
Dialect-dependent leaves	44.3%
Number of reordered nodes	29137
Number of virtual leaves	7085

Table 3.77: Reordered Tree statistics for GALE+TRANSTAC setup.

3.6.5.5.3. Experiments

We use our vowelized Arabic model as a baseline in our experiments. The vocabulary has about 737K words, and 2.5M pronunciations. The language model is a 4-gram LM with 55M n-grams. Speaker adaptation includes VTLN and FMLLR. All models are ML trained. In addition to the GALE EVAL-06 and DEV-07 test sets, we also used a TRANSTAC test set comprising two hours of audio. The dialect labels are derived from the audio file names. This information may be used under the rules of the GALE evaluation. The file names encode TV channel and program information. Please note also that this information is available for deployed systems that commercialize GALE technology, such as IBM's TALES system.

¹² This can be done via a separate dialect ID tool, selecting the dialect with the best likelihood, or other sources of information.

Acoustic Model	Regular tree	Dialect tree
2330h GALE	19.2%	18.6%
+ 500h TRANSTAC	19.6%	18.7%

Table 3.78: Comparison of regular tree and tree with dialect questions. GALE DEV-07 test set.

In the first experiment we train four acoustic models. Each model has 8K states and 400K Gaussians. The models are trained on either 2330 hours of GALE data or on the GALE data plus 500 hours of TRANSTAC data. For each training set, we train one model using the regular (phonetic context only) question set and one using phonetic and dialect questions. The test set is DEV-07. The results are summarized in Table 3.78. For the GALE model, we see an improvement of 0.6% WER. The improvement for the GALE+TRANSTAC training set is slightly higher, 0.9%. The results suggest that the decision tree with dialect questions can better cope with diverse, and potentially mismatched, training data.

In the second experiment (Table 3.79), we use the same set of acoustic models as before, but the vocabulary and language model are now TRANSTAC specific and the test set is drawn from TRANSTAC data. Adding TRANSTAC data improves the error rate from 35.9% to 25.9%. Adding the dialect-specific questions to the tree building process improves the error rate by an additional 1.2%. We did not test the dialect tree trained on GALE data only. The tree does not contain any TRANSTAC related questions, since the models were trained on GALE data only.

Acoustic Model	Regular tree	Dialect tree
2330h GALE	35.9%	-
+ 500h TRANSTAC	25.9%	24.7%

Table 3.79: Comparison of regular tree and tree with dialect questions. TRANSTAC test set.

Test set	Regular tree	Dialect tree
EVAL-06	30.2%	29.6%
DEV-07	20.3%	19.5%

Table 3.80: Comparison of regular tree and tree with dialect questions with unsupervised training data. GALE EVAL-06 and DEV-07 test sets.

In the final experiment, we use a large amount of unsupervised training data from our internal TALES data collection (see Section 3.6.5.2.1 for a description of the data, and Section 3.6.5.3 for unsupervised training experiments). The acoustic model was trained on 2330 hours of GALE data plus 5600 hours of unsupervised training data. The results are shown in Table 3.80. The dialect-dependent decision tree reduces the error rate by 0.6% on EVAL-06 and 0.8% on DEV-07. While adding more unsupervised training data does not help if large amounts of supervised training data are available (Table 3.69), we observe that the dialect tree is able to compensate for adding “unuseful” data (Table 3.80).

3.6.5.6 Conclusion

We have presented a set of techniques for Arabic broadcast transcription that, taken together, lead to word error rates below 10%. Techniques that contribute to this level of performance include the use of a very large recognition vocabulary, cross-adaptation between unvowelized and vowelized acoustic models, and flat start training to automatically vowelize Arabic audio. We have also presented additional results on large-scale, unsupervised discriminative training, and on the modeling of Arabic dialects using dialect questions in the decision tree.

3.6.6 Recognition Units in Arabic Automatic Speech Recognition

Authors: Long Nguyen, Tim Ng, Kham Nguyen, Rabih Zbib, and John Makhoul

3.6.6.1 Introduction

Arabic presents two primary difficulties to automatic speech recognition (ASR) systems, both due to the Arabic writing system. The first difficulty is the morphological complexity that arises from the attachment of affixes, which renders recognition vocabularies very large if one wants to minimize out-of-vocabulary (OOV) words. For example, a 65K Arabic vocabulary typically has an OOV rate of 5%, compared to 0.5% for the same size English vocabulary. The second difficulty arises due to the fact that the short vowels are typically not written. Both difficulties are partially overcome by the use of automatic morphological analyzers, which also produce vocalizations, but the resulting analysis and vocalization can be errorful at times.

3.6.6.2 System Description

We describe the use of either words or morphemes as recognition units, and the use of either graphemes or phonemes as *phonetic* units in Arabic ASR to handle the two key problems mentioned previously. Specifically, we designed four Arabic ASR systems: two word-based systems and two morpheme-based systems. In the first word-based system, each word is modeled by one or more sequences of phonemes of its phonetic pronunciations. In the second word-based system, each word is modeled by exactly one sequence of letters of its spelling. The third and fourth systems use morphemes as recognition units. Morphemes are determined by either a simple morphological decomposition using a small set of affixes and a few rules or by an elaborate linguistically-driven morphological analyzer. Each morpheme is modeled by sequences of phonemes of its pronunciations derived from the corresponding word's pronunciations during the decomposition process.

Experimental results using these four systems (developed for the GALE Phase 3 evaluation) show that they have comparable state-of-the-art performance individually. However, they all seem to complement each other quite well such that the combination of the four systems using ROVER provides substantial improvement in performance when compared to each single system. At the core of each ASR system is the Byblos multi-pass

recognizer. Various acoustic and language models at different levels of sophistication are deployed at different passes and/or stages.

3.6.6.2.1. Recognizer

The Byblos multi-pass recognizer (Nguyen and Schwartz 1997) first performs a fast match of the data to produce scores for numerous word endings using a coarse state-tied-mixture (STM) acoustic model (AM) and a bi-gram language model (LM). Next, a state-clustered tied-mixture (SCTM) AM and an approximate trigram LM are used to generate lattices. Lattices are then re-scored using a crossword SCTM AM and a 4-gram LM. The best path of the rescored lattice is the recognition result. In other words, the decoding process is a three-step sequence (fast-match, lattice generation, and lattice rescoring) with finer-detailed models being used on narrower search space at later steps (Nguyen *et al.* 2002).

The decoding process is repeated three times. First, speaker-independent (SI) acoustic models are used in the decoding to generate hypotheses for unsupervised adaptation. Then, the decoding is repeated but with speaker-adaptively-trained (SAT) acoustic models that have been adapted to the hypotheses generated in the first stage. The last decoding is similar to the second, but acoustic models are adapted to the second stage's hypotheses using a larger number of regression classes.

3.6.6.2.2. Acoustic Model Training

The typical procedure for training acoustic models at BBN can be logically grouped into these four sequential stages.

Frontend Processing: 14-dimensional Perceptual Linear Predictive (Hermansky 1990) cepstral coefficients are extracted from the overlapping frames of audio data with a frame rate of 10ms. Cepstral mean subtraction is applied for normalization. The normalized energy is used as the 15th component. Nine successive 15-component frames centered at the current frame are concatenated and then reduced to a 60-dimensional feature vector using Linear Discriminant Analysis (LDA) and decorrelated using Maximum Likelihood Linear Transformation (MLLT) (Zhang *et al.* 2004). The dimension reduction is applied differently for SI and SAT models.

ML-SI Training: The SI AMs are trained using the Maximum-Likelihood (ML) criterion. Feature dimension reduction is done via global LDA and MLLT transformations estimated from and applied to all training data. These models are to be used in the SI decoding stage.

ML-SAT Training: The SAT AMs are then trained using also the ML criterion. Each 15-dimensional feature vector is transformed using a speaker-cluster-dependent Constraint Maximum Likelihood Linear Regression (CMLLR) transform (Gales 1997) before the concatenation of nine successive frames for dimension reduction. Both LDA and MLLT transformations are speaker-cluster-specific at this stage. The resulting 60-dimensional feature vectors are further transformed using new CMLLR transformations. These models are used only in the adapted decoding stages.

MPFE Training: In the last stage of acoustic model training, all training data is decoded using the ML models to generate lattices. Then a new set of AMs, both SI and SAT, are estimated using these lattices under the Minimum Phoneme-Frame Error criterion (Zheng and Stolcke 2005).

The total amount of acoustic training data used in this effort is about 1400 hours selected through light supervision (Nguyen and Xiang 2004) of all data available to the GALE community. Specifically, the data consist of the following corpora: BBN-FBIS (43hr), TDT4 (67hr), Iraqi Dialect (50hr), GALE Year 1 (144hr), Phase 2 (603hr), and Phase 3 (526hr).

3.6.6.2.3. Language Model Training

The language models were estimated through interpolation of several sub-*N*-Gram models trained on disjoint subsets of the 1.7 billion words of Arabic text data available to the GALE community as of May 2008. Each individual sub-*N*-Gram model was trained using the modified Kneser-Ney smoothing technique.

3.6.6.3 Implementation

Arabic Text Normalization: Due to writing conventions in Arabic, sometimes the same word has different written forms. This is especially true for words that start with the letter “hamza” (corresponding to the glotal stop). So, for consistency in lexical representation for ASR, we map all different forms of “hamza” (“<“, “>“, and “|” using Buckwalter transliteration scheme) at the beginning of the word, or after the common Arabic prefixes “Al” and “w”, to “alif” (“A”). Also, for certain frequent words, we map the “alif maksura” (“Y”) at the end of the word to “yeh” (“y”) or vice versa. However, for scoring of ASR output, all forms of “hamza” are equated to “alif” without the restriction above.

Master Phonetic Dictionary: As described by Afify *et al.* (2005), Xiang *et al.* (2006) and Ng *et al.* (2009), we constructed phonetic dictionaries for Arabic by using the Buckwalter morphological analyzer and manually-vocalized corpora.¹³ Minor changes to this procedure include the addition of extra affixes used in major Arabic dialects to the list of Modern Standard Arabic (MSA) affixes used in the Buckwalter Analyzer. We also added all vocalizations found in the manually-vocalized corpora (LDC’s Arabic TreeBanks, LDC’s Iraqi dialect lexicon, and the 45-hour BBN-FBIS corpus). As a result, our Arabic master phonetic dictionary, as of now, consists of about 1.2 million words, each with approximately 3.76 pronunciations on average. This is the master dictionary from which we derive phonetic pronunciations for all phonetic word-based and morpheme-based Arabic ASR systems.

¹³ <http://www.qamus.org/morphology.htm>

3.6.6.3.1. Phonetic System (P)

The design of our word-based phonetic system (P) is a straightforward implementation of a typical ASR system. The recognition units are Arabic words and each word is modeled by one or more sequences of phonemes of its pronunciation(s). The largest AM (i.e., the crossword SCTM model) has about 220K crossword quinphone *states* sharing 7K sets of Gaussians (or codebooks). Note that we use two level Gaussian mixture models with several different states having separate sets of mixture weights, but sharing the same set of Gaussians (or codebooks, as described in detail by Nguyen *et al.* (2002)). The total number of Gaussians in this model is about 900K. The exact values for all four systems can be found in Table 3.81.

Sys.	#states	#codebooks	#Gaussians
P	220694	6882	889146
G	212066	6580	867707
M1	222506	6829	883117
M2	220561	6854	885518

Table 3.81: AM comparison: all four systems use about the same number of model parameters.

The recognition vocabulary used in this phonetic system consists of 390K words. Each word has 4.03 pronunciations on average. This is a subset of the 490K words that occur at least 30 times in the language model training data or at least 3 times in the transcripts of the acoustic training data. That means 100K words in the list of the 490K most frequent words do not exist in our 1.2M-word master dictionary, because they could not be analyzed by the Buckwalter Analyzer, and they do not occur in the manually-vocalized corpora. This system's language model consists of 137M trigrams (pruned based on entropy) used for decoding and 585M (unpruned) 4-grams used for lattice rescoring. The OOV rate of this vocabulary and the size of the LMs of this system are compared against those of other systems' in Table 3.82 and Table 3.83.

Sys.	vocab.	pron.	#3-grams	#4-grams
P	390K	4.03	136,767,904	585,331,357
G	490K	1.00	140,915,143	617,448,176
M1	289K	4.43	137,586,807	565,474,893
M2	284K	3.69	142,068,884	563,803,750

Table 3.82: LM comparison: all four systems have about the same number of n-grams even though the size of the morpheme vocabularies is substantially smaller than that of the word vocabularies.

Sys.	#units	at6	ad6	dev07	eval07	dev08
P	390K	4.02	3.39	4.36	2.88	1.44
G	490K	3.43	2.86	3.78	2.07	0.84
M1	289K	1.31	1.20	2.82	1.89	0.94
M2	284K	0.66	0.68	0.81	0.66	0.56

Table 3.83: OOV rate comparison: morpheme-based vocabularies have better coverage than word-based vocabularies for Arabic.

3.6.6.3.2. Graphemic System (G)

The word-based graphemic system (G) is similar to the phonetic system except that it doesn't use phonetic pronunciations and it has a much larger recognition vocabulary. The recognition units are also Arabic words, but each word is modeled by exactly one sequence of letters of its spelling. The largest AM of this system has about 210K states sharing 7K sets of Gaussians for a total of about 900K Gaussians (as shown in row G of Table 1). Since this system doesn't depend on the real phonetic dictionary, its recognition vocabulary uses all 490K most frequent words. Its LM consists of 140M (pruned) trigrams and 617M (unpruned) 4-grams.

3.6.6.3.3. Morphemic System 1 (M1)

The morphemic system 1 (M1) is a morpheme-based phonetic system. Morphemes are determined by a simple morphological decomposition of the words without their context using a small set of affixes and a few rules. More details on various morphological decomposition strategies and their effect to ASR performance for morphemic systems developed in the past can be described by Xiang *et al.*(2006). In the current M1 system, the decomposition process uses 12 prefixes and 34 suffixes. The prefixes are: *Al, bAl, fAl, kAl, ll, wAl, b, f, k, l, s, w*, and the suffixes are: *An, h, hA, hm, hmA, hn, k, km, kn, nA, ny, t th, thA, thm, thmA, thn, tk, tkm, tm, tnA, tny, tynA, wA, wh, whA, whm, wk, wkm, wn, wnA, wny, y, yn*. The process also utilizes a list of 128K most frequent decomposable words that should not be decomposed (hereafter referred to as a *blacklist*) since it has been shown that this really improved recognition performance. For a candidate word to be decomposed, it has to satisfy the following five conditions: (1) it doesn't belong to the blacklist, (2) it consists of at least one of the pre-determined affixes, (3) its decomposed affixes' pronunciations match the pre-determined pronunciations, (4) its stem must exist in the master dictionary, and (5) its stem must be at least two letters long.

As shown in Table 3.81, the acoustic models of this system were designed to have about the same number of system parameters as all other systems. However, the recognition units, i.e., the *morphemes*, in this system consist of 289K morphemes obtained through the decomposition of all 490K most frequent words, including its blacklist, and all words occurring in the transcripts of the acoustic training data. Its LM also has about the same number of n-grams used in the other systems, as shown in Table 3.82.

3.6.6.3.4. Morphemic System 2 (M2)

The morphemic system 2 (M2) is another morpheme-based phonetic system, but the morphemes in this case were determined by a more sophisticated decomposition process. Briefly, the process consists of two steps. First, we use an Arabic morphological analyzer to decompose all AM and LM training data consisting of about two billion words. Each word, if occurring more than once, can be decomposed into different sequences of morphemes depending on its different contexts. Note that each word instance is decomposed into exactly one sequence of morphemes of the form *[prefix] + stem +*

[*suffix*], where either the prefix or the suffix or both can be missing. Then, we collect all the decomposed forms of the 490K most frequent words and all words occurring in the acoustic training data and pass them through the final decomposition process. Similar to the construction of the recognition units in system M1, we also utilize a blacklist of the 128K most frequent decomposable words. The decomposable words are finally decomposed as follows:

- M. if the word is in the blacklist, keep unchanged;
- N. else if no *prefix*, decompose into *stem* and *suffix*;
- O. else if no *suffix*, decompose into *prefix* and *stem*;
- P. else if *prefix+stem* is in the blacklist, decompose into *prefix+stem* and *suffix*;
- Q. else if *stem+suffix* is in the blacklist, decompose into *prefix* and *stem+suffix*;
- R. else decompose into *prefix*, *stem*, and *suffix*.

This decomposition process produced about 284K morphemes to be used in the recognition vocabulary in this system. Both the AMs and LMs of this system have about the same sizes compared to those of other systems, as shown in Table 3.81 and Table 3.82. A full description of this system and a comparison to the M1 morphemic system is provided by Ng *et al.* (2009).

3.6.6.4 Experimental Results

Development Test Sets: To support the research and development of these four systems, we used five Arabic test sets. The first two test sets, **at6** and **ad6**, were constructed at BBN at the start of the GALE program. Each is about six hours long, and uses Arabic broadcast programs aired in November 2005 and January 2006. These two test sets contain long segments of broadcast *stretches* of about ten to twenty minutes long. The remaining three test sets are development or evaluation test sets designed by LDC/NIST to be used by all GALE participants. Each of these three test sets is about three hours long. They consist of Arabic broadcast programs aired in November 2006 (**dev07**), December 2006 (**eval07**), and May 2007 (**dev08**). These three test sets contain only short *snippets* ranging from two to four minutes long (to facilitate the research and evaluation of the down-stream machine translation task of the GALE program). Each test set includes both *broadcast news* and *broadcast conversations* with almost equal amounts.

3.6.6.4.1. OOV Comparison

As shown in Table 3.83, both word-based systems (P and G) have high OOV rate for all the test sets even though they use rather large vocabulary (400-500K words). Except for **dev08**, the OOV rates are 3% or 4% for all other test sets. The graphemic system (G) has a bit lower OOV rate since it uses a larger vocabulary. In contrast to using words as recognition units, both morphemic systems (M1 and M2) have much lower OOV rate even though their vocabulary sizes are substantially smaller. It is interesting to point out that the vocabulary of the M2 system has similar OOV rates across all test sets. Note that we scaled the morpheme OOV rate of the morpheme vocabulary by the

decomposition rate (number of morphemes after decomposition divided by number of words before decomposition) to make it comparable to the standard word OOV rate.

3.6.6.4.2. Individual Results

Sys.	at6	ad6	dev07	eval07	dev08
P	18.8	16.9	10.6	11.6	12.1
G	18.5	16.7	11.6	12.2	12.5
M1	18.1	17.1	10.3	11.1	11.6
M2	17.6	16.3	10.2	10.8	11.8

Table 3.84: Individual system results: the simplest system (G) has the overall highest WER and the most sophisticated system (M2) has the lowest WER.

Table 3.84 shows the WER of the four systems on the five test sets. Overall, systems using morphemes as recognition units perform better than systems using words as recognition units. Using morphemes derived from the linguistically-driven approach in M2 produces the lowest WER among the four systems. Between the two morphemic systems, M2 tends to be more robust in terms of both OOV rate and WER. M2 is clearly the best system on the two BBN-constructed test sets (at6 and ad6). Between the two word-based systems, the phonetic system is better than the graphemic system for the three GALE test sets (dev07, eval07, and dev08). However, the graphemic system is better than the phonetic system on the two BBN test sets (at6, and ad6). It's important to point out that the two BBN test sets are much more difficult than the three LDC/NIST test sets based on WER.

These four sets of results seem to correlate well with the complexity of the design of an Arabic ASR system. The simplest system has the highest WER (G) and the most sophisticated system (M2) has the lowest WER among the four systems. This finding would probably provide sufficient information for a designer of an Arabic ASR system to decide the type of recognition units to use to achieve the best tradeoff required by his/her application. For example, even though it has the worst performance among the four systems, the word-based graphemic approach is clearly the simplest and easiest to implement, based on the fact that it doesn't require a phonetic dictionary. Furthermore, each word would have exactly only one *pronunciation* and, consequently, the recognizer would run faster in comparison to phonetic systems that have four pronunciations per word on average. However, if accuracy is the top requirement, the designer must be prepared to construct a phonetic dictionary with good quality and coverage. Also, a decent morphological analyzer that can decompose words within context into morphemes is required.

3.6.6.4.3. System Combination Results

In addition to the scientific curiosity to search for the optimal recognition unit for Arabic ASR, another goal of this work is to have diverse, but complementary systems to be used within the ROVER system combination framework (Fiscus 1997). As shown in Table 3.85, the combination of all four systems (P+G+M1+M2) provides the lowest WER for all test sets. The relative reduction in WER for all five test sets in comparison the best system (M2) is from five to ten percent. Various combinations of two or three

systems, starting with the phonetic system (P), are shown in the top two blocks of Table 5. Generally, combining two systems produces about zero to three percent relative reduction in WER. Combining three systems increases the relative WER reduction further to about three to six percent.

ROVER	at6	ad6	dev07	eval07	dev08
P+G	17.4	15.8	10.5	10.9	11.6
P+M1	17.7	16.4	10.1	10.9	11.4
P+M2	17.3	15.9	10.2	10.7	11.5
P+G+M1	16.9	15.6	9.9	10.6	11.0
P+G+M2	16.7	15.4	9.8	10.4	11.0
P+M1+M2	17.2	15.8	9.8	10.5	11.1
P+G+M1+M2	16.6	15.3	9.7	10.3	10.8

Table 3.85: Various system combination results: combining two systems (P+*) produces 0-3% in WER reduction relative the best single system (M2); combining three systems (P+*+*), 3-6%; and the lowest WER obtained by combing all four systems.

3.6.6.5 Conclusion

We have described two possible types of recognition units, either words or morphemes, and the use of either graphemes or phonemes as *phonetic* units for Arabic automatic speech recognition. The study involves implementation of four different Arabic ASR systems. The results of these four systems show that they all have comparable state-of-the-art performance. They also show a good correlation to the complexity of the design of an Arabic ASR system: the simplest system; the word-based graphemic system, has the highest WER; and the most sophisticated system; the linguistically-driven morphemic system, has the lowest WER. In addition, even though all four systems use the same underlying ASR technology, the use of different recognition units seem to be quite complementary to each other such that the combination of the four systems produces substantially-improved combined results.

3.6.7. Development of the SRI/NIGHTINGALE Arabic ASR System

Authors: Dimitra Vergyri, Arindam Mandal, Wen Wang, Andreas Stolcke, Jing Zheng, Martin Graciarena, David Rybach, Christian Gollan, Ralf Schlüter, Katrin Kirchhoff, Arlo Faria, and Nelson Morgan

3.6.7.1 Introduction

In order to achieve the high performance speech translation targets set by the GALE program, it has been necessary to drastically improve the accuracy of the first stage of the systems, which involve automatic speech recognition (ASR) in the source language. In this work we focus on the MSA ASR system developed for broadcast news (BN) and broadcast conversations (BC).

The morphological complexity of Arabic and other language features, like the lack of short vowels in standard orthography, introduce new requirements for the design of the ASR systems. The characteristics of the Arabic language and the problems they pose for ASR are described in detail by Kirchhoff *et al.* (2003). Previous work that dealt with

MSA ASR includes that of Afify *et al.* (2005), Messaoudi *et al.* (2006) and Soltau *et al.* (2007).

This section describes the following aspects of system design, and reports their impact on the system's performance: data preprocessing, lexicon and pronunciation probability generation, training data size, vocabulary size, sentence segmentation, front end features and discriminative training for acoustic models, and system architecture including system combination and cross adaptation. Our system includes components from two recognition engines: DecipherTM from SRI and the speech recognition system from RWTH Aachen. Results are reported for various GALE datasets that include both BN and BC speech.

3.6.7.2 Data

For acoustic training we used the 16 kHz Arabic data distributed by the Linguistic Data Consortium (LDC) with quick transcriptions: 45 hours of FBIS, 85 hours of TDT-4, 580 hours of GALE BC, and 740 hours of GALE BN data.¹⁴ We experimented with unsupervised training on an additional 3000 hours of untranscribed data, but these experiments showed ultimately no gains and are not included here. We also excluded all data sampled at 8 kHz that were distributed by LDC, since experiments showed no benefit from its use. All acoustic data was aligned with the transcriptions using the flexible alignment procedure described by Venkataraman *et al.* (2000). Long silences and regions with poor alignment confidence were discarded. Eventually, a total of 1120 hours was selected for training the acoustic models.

The language model (LM) data was also provided mostly by LDC, but some data came from other sources (under GALE). The total text used for LM included about 1.1 billion words: 490M of Arabic Gigaword (LDC, 2nd edition), 86M UN-data (LDC), 79M GALE data (LDC), 67 M CMU Web news data and 410M Web news data (collected by Cambridge University covering the dates between 11/2000 and 10/2006).

We evaluated the results on three development sets: the 2006 evaluation set (eval06: 3 hours, 11481 BN and 11243 BC words), the 2007 development set (dev07: 2.5 hours, 11742 BN and 6495 BC words) and the 2007 evaluation set (eval07: 4 hours, 14355 BN and 14187 BC words).

3.6.7.3 Text Processing and Lexicon Generation

3.6.7.3.1. Text Data Processing

All text data was processed using the MADA tool from Columbia University (Habash and Owen 2005), which uses a statistical morphological tagger to fix common errors or ambiguities in the Arabic orthography, like the presence of Hamzas above or below the letters Alef, Yeh and Waw, and the marking of final Yeh and Teh Marbuta.¹⁵ MADA also provides full word diacritization, which was used as a means for developing the pronunciation lexicon. Other post-processing scripts were used to expand numbers

¹⁴ Data released up to Oct 2007 were used in this work.

¹⁵ http://www1.cs.columbia.edu/~rambow/software-downloads/MADA_Distribution.html

and dates in the text data to words, remove nonlexical tags and punctuation marks, and normalize the orthography of some frequent names.

Text Processing	Eval06	Dev07
MADA_1	36.3	25.7
MADA_2	35.7	25.2
No pron. probs	36.8	26.0
With pron. probs	35.7	25.2

Table 3.86: Effect of improved text processing on WER.

We did not compare our data processing method with that of other GALE sites (e.g., some sites do full normalization of Alefs and final Yeh), but we found that improved text processing had a significant effect on the accuracy of our models. Table 3.86 compares the word error rate (WER) results using MADA_1 (scripts released in 2005) versus MADA_2 (released in 2007 as v1.8). MADA_1 was based on the Buckwalter Arabic Morphological Analyzer (BAMA) v.1 while MADA_2 used BAMA v.2, and the tagger was trained on more training data.¹⁶ The acoustic models used for this comparison were maximum likelihood (ML), speaker independent (SI) MFCC models trained on 450h. The bigram LM used a 400K word vocabulary and was trained on ~500M words of the data released before 2007.

3.6.7.3.2. Lexicon

Since standard (undiacritized) Arabic orthography lacks the symbols that indicate the position of short vowels and the lengthening (doubling) of consonants (Kirchhoff *et al.* 2003), all Arabic ASR systems have the problem of representing these in the pronunciation lexicon (Afify *et al.* 2005; Messaoudi *et al.* 2006; Soltau *et al.* 2007). In our system we used the diacritized MADA output as the basis of our pronunciation lexicon. The mapping between the diacritized orthography and the phonemic representation is fairly straightforward. It is mostly a 1 - 1 mapping with a few rules that describe the pronunciation of Arabic (e.g., the Teh Marbuta is pronounced “at”).

For acoustic model training the MADA-diacritized word form was used with a single pronunciation per word. We found that this improved the acoustic models by about 0.5%, compared to using undiacritized orthography and multiple word pronunciations. For the test lexicon we used the undiacritized orthography, and all diacritizations found for each word in the training data were used as pronunciation variants. This procedure resulted in an average of 3.3 pronunciations per word, for the words that MADA could diacritize. MADA failed to produce output for about 3% of training word tokens, which account for about 20% of the total training data word types (mostly infrequent words). For these words we used an automatic pronunciation generation model that was trained on the rest of the pronunciation lexicon to obtain their pronunciation, as is described by Bisani and Ney (2003).

The pronunciation probabilities for the words appearing in the acoustic training data were obtained using acoustic forced alignment of the data with the phonetic

¹⁶ LDC distributions LDC2002L49 and LDC2004L02

pronunciations, and calculating smoothed empirical frequencies of the pronunciations of each word. For rare words or words not found in the acoustic training data, we obtained the probability for each pronunciation using a phone n-gram model trained on the forced alignment phone sequences. The n-gram probabilities were scaled and normalized. The effect of pronunciation probabilities in the lexicon used to generate the first-pass ASR output is shown in rows 4 and 5 of Table 3.86. The same acoustic models were used as for the experiments in upper rows of Table 3.86.

3.6.7.4 Language Modeling

3.6.7.4.1. Language Model Training

The LM was trained using the SRILM toolkit (Stolcke 2002) with the undiacritized MADA-processed data. We trained source-specific LMs, each with modified Kneser-Ney smoothing, which were then interpolated for the final LM. The training sources were FBIS, TDT-4, GALE BN and BC transcripts (manual transcriptions and web data), Arabic Gigaword, CMU web data, CU web data, and UN data, for a total of about 1.1B words. Interpolation weights were tuned on a balanced heldout mixture of GALE BN and BC manual transcripts, of about 300K words. The final LM used a 600K vocabulary and was entropy-pruned to 40M bigrams for lattice generation decoding. For rescoring purposes a much bigger 4-gram was used with about 100M bigrams, 73M trigrams, and 73M 4-grams. These LMs were used in both the SRI and the RWTH systems.

We also experimented with morphological-class LMs and factored LMs, but as more data became available the improvement from these models diminished, and they were not used in the final system.

3.6.7.4.2. Vocabulary Selection

To deal with the high morphological variability of Arabic, we had to enlarge the vocabulary far beyond what is typically used for English ASR. We used the approach described by Venkataraman and Wang (2003) to choose the vocabulary, using the same held-out set as for LM tuning. Table 3.87 shows the effect of vocabulary size on OOV (out-of-vocabulary) rate and WER, using all the available data for LM.

Vocab. Size	OOV(%)		WER(%) 2gram/4gram	
	Eval06	Dev07	Eval06	Dev07
64K	5.36	3.88	36.4/33.6	25.3/22.5
128K	3.09	2.05	35.8/32.7	24.6/21.7
256K	2.10	1.34	35.6/32.5	24.5/21.6
500K	1.20	0.69	35.4/32.4	24.3/21.5
600K	1.07	0.58	35.5/32.5	24.3/21.6

Table 3.87: Effect of vocabulary size on OOV and WER

3.6.7.4.3. Effect of LM Training Data Size

Using about half the available LM data (about 500M words from Gigaword, UN and the first two years of GALE) we found that the performance of the LM (after 4-gram

rescoring) degrades about 1-1.5% absolute on various test sets. The experiment was performed using a 500K vocabulary.

3.6.7.5 Acoustic Modeling

3.6.7.5.1. SRI Acoustic Models

As described in Section 3.6.7.6.3, the SRI system uses multiple models at different steps: non-cross-word (non-cw) models are used for first-pass lattice generation, and for lattice rescoring. Cross-word (cw) models are used only for lattice rescoring and N-best list generation.

Frontends

We used an MFCC front end with 16 kHz sampling rate and a PLP one with 8 kHz sampling rate, to help de-correlate errors for system combination. (The 8-kHz frontend is motivated by the fact that some of the BC data is narrowband and/or noisy.) The MFCCs were augmented with multilayer perceptron (MLP) features as described by Zhu *et al.* (2005). The non-cw MFCC and the PLP frontends have 13 cepstral coefficients, while the cw MFCC has 19. They all use 1st, 2nd and 3rd order derivatives, with HLDA for reduction to 39-dimensional feature vectors. Table 3.88 shows the performance of the various models on different test sets. The PLP model without MLP features does not perform as well as the MFCC models, but because of the 8-kHz front end it performs well on the BC part of the data.

	<i>Eval06</i>			<i>Dev07</i>		
	<i>BC</i>	<i>BN</i>	<i>all</i>	<i>BC</i>	<i>BN</i>	<i>all</i>
ML models						
Non-cw MFCC+MLP	33.6	21.8	27.6	19.7	11.3	14.3
cw PLP	32.1	21.4	26.7	18.8	12.0	14.4
cw MFCC19+MLP	31.8	21.1	26.4	18.2	10.9	13.5
MPFE models						
Non-cw MFCC+MLP	30.9	20.8	25.8	18.4	10.5	13.3
cw PLP	30.5	21.1	25.7	18.6	11.2	13.8
cw MFCC19+MLP	29.8	19.7	24.7	16.5	9.8	12.2
cw PLP+fMPE	30.1	20.4	25.2	17.8	10.8	13.3

Table 3.88: Results with different front-ends used at SRI acoustic models, contrasting ML and discriminative training conditions. All results are obtained with SI models after rescoring 4gram lattices created with the same non-cw MFCC model.

Discriminative Training

All models were trained with discriminative minimum phone frame error training (MPFE) (Zheng and Stolcke 2005). We can see the gains over ML training in Table 3.88. We also experimented with using feature-level minimum phone error (fMPE) training (Povey *et al.* 2005; Zheng *et al.* 2007). We found that the MLP features, which are already discriminatively trained, do not combine well with fMPE (there was practically no improvement from fMPE). That was the reason we decided not to include MLP features for the PLP frontend and to use fMPE instead. We see that the PLP+fMPE

models are comparable to the MFCC+MLP non-cw models after discriminative training. The crossword MFCC19+MLP models performed the best.

Gender Dependent Models

Since we did not have data annotated for speaker gender, we used a gender prediction GMM trained on English data, to predict speaker gender in both training and test data. Then we performed MPE-MAP adaptation on the final discriminatively trained models, using the single-gender data only. This gave about 0.3% absolute improvement for individual models.

Duration Models

We used the approach described by Gadde (2000) to train duration models for each acoustic model in our system. Single multivariate Gaussians are used to model phone frame durations in words and triphones, normalized for a speaker's average speaking rate. We employ these duration models for rescoring all N-best lists used in system combination (see Section 3.6.7.6.3). On average, duration modeling contributed a relative gain of ~2.5% for each component system. The improvement in the final word error was about 0.2% absolute.

3.6.7.5.2. RWTH Acoustic Models

The RWTH system consists of two subsystems using two different acoustic models that share the same acoustic front end. MFCCs (16 cepstral coefficients) normalized by cepstral mean and variance normalization are augmented with a voicing feature. An LDA matrix projects the concatenation of 9 consecutive feature vectors in a sliding window to 45 components. As in the SRI MFCC models, this reduced feature vector is augmented with phoneme posterior features estimated by a neural network (Hoffmeister *et al.* 2007). The two acoustic models used differ in their treatment of crossword context; one subsystem uses within-word models only. In training, speaker variations are compensated by applying VTLN to the MFCC filter bank and speaker adaptive training (SAT) based on CMLLR. The models were enhanced by performing discriminative training with the MPE criterion.

3.6.7.6 System Architecture

3.6.7.6.1. Acoustic Segmentation

Acoustic segmentation was applied in tasks where no manual transcription was available. Our segmentation method operates on the output of a speech recognizer. Several segment features are used to optimize the complete segmentation of a recording. This method is described in detail by Rybach *et al.* (2007). In experiments where we used the non-cw system without LM rescoring and cross adaptation, we achieved a WER of 16.1% on Dev07 using the described segmentation. This result is quite close to the 15.9% achieved when using manual segmentations. However, there is still potential for improvement, which becomes apparent when splitting the manual segmentation before and after overlapping speech, yielding an error rate of 15.4%.

3.6.7.6.2. RWTH ASR Architecture

	non-cw system	cw system
Pass 1	18.6	-
Pass 2	14.4	14.4
Pass 3	13.9	14.2
Weight Optimization and Combination	12.7	

Table 3.89: Intermediate results of the RWTH subsystems on the dev07 corpus.

Speaker-independent within-word models are used in the initial recognition pass. The output of this system is used to estimate the text-dependent CMLLR transforms for both subsystems. Segment clusters obtained by generalized likelihood ratio clustering with a BIC-based stopping criterion act as speaker labels for the speaker adaptation methods applied. The second recognition pass is carried out using the CMLLR transformed features and acoustic models trained with SAT. The lattices produced by the two systems are rescored using the full 4-gram language model, while a pruned bigram LM is used for generating the lattices. The acoustic models of both systems are adapted to this rescored output using MLLR on the means. We used a cross-adaptation scheme to benefit from having two subsystems. The results of the final recognition pass are again rescored using the larger LM. Eventually, the lattices produced are combined by a method based on min-fWER decoding (Hoffmeister *et al.* 2007). Intermediate results for both systems are given in Table 3.89.

3.6.7.6.3. SRI ASR Architecture

Figure 3.37 illustrates the design of the SRI ASR system. We generated lattices with the non-cw model twice (before and after adaptation) and cross-adapted all models used for final lattice rescoreing to outputs of the other systems. The best system (MFCC19+MLP) was cross-adapted to the ROVER combination of the other two systems. The results for different stages of the system are given in Table 3.90.

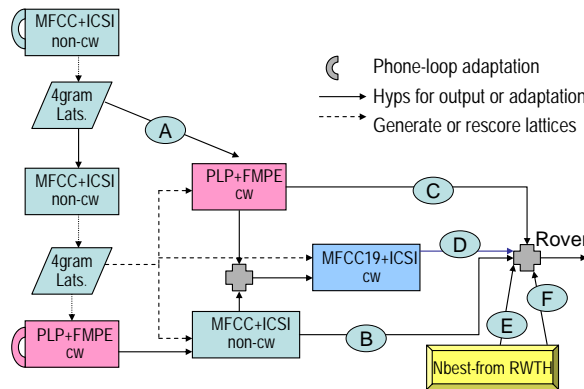


Figure 3.37: SRI System Architecture

Output	Eval06 (manual seg.)			Dev07 (manual seg.)		
	<i>BN</i>	<i>BC</i>	<i>all</i>	<i>BN</i>	<i>BC</i>	<i>All</i>
<i>A</i>	20.7	33.1	26.7	10.7	19.1	15.3
<i>B</i>	19.5	31.2	25.2	9.7	16.5	12.2
<i>C</i>	19.3	31.1	25.1	9.8	15.9	12.0
<i>D</i>	19.1	29.9	24.4	9.3	15.2	11.4
<i>E-noncw</i>	19.8	31.1	25.4	10.6	16.7	12.7
<i>F-cw</i>	20.4	31.8	26.0	10.6	16.7	12.8
ROVER3	18.8	29.0	23.8	9.2	14.7	11.1
ROVER5	18.2	27.9	23.0	8.9	14.3	10.8

Table 3.90: WER results at different system stages. ROVER3 includes SRI-only systems B, C and D, while ROVER5 includes also the two RWTH systems E and F.

3.6.7.6.4. System Combination

We used N-best ROVER on the three N-best lists obtained from the SRI system, and the two N-best lists from RWTH, to obtain the final output. The results are shown in the last line of Table 3.89 for the Eval06 and Dev07 test sets, while the final results for the latest Eval07 test are shown in Table 3.91.

Output	Eval07			
	Manual seg.		Automatic seg.	
	<i>BN</i>	<i>BN</i>	<i>BC</i>	<i>all</i>
ROVER3	10.3	10.1	16.0	13.0
ROVER5	9.8	9.9	15.5	12.7

Table 3.91: The final system WER on Eval07. Manual segments were available only for the BN part.

Cross-adaptation schemes were also explored where the output from an intermediate pass from RWTH was used to adapt the SRI models, but since the SRI system was already using cross-adaptation among fairly different systems, cross-site adaptation did not give any further improvements.

3.6.7.7 Discussion and Conclusions

We have described the development process of a large vocabulary ASR system for MSA. The system was based on our English ASR experience, but we paid special attention to lexicon development and increased vocabulary size to deal with the needs of the new language. The different models had different enough features to achieve good cross-adaptation and system combination results. Our system obtained a WER of about 10% on BN and 15-25% on BC on different test sets. Since the goal of the GALE project is speech-translation performance, we compared the machine translation (MT) results obtained using the output of our system to those obtained using the reference hypotheses and found a difference of less than 0.5 point of BLEU or 1.5 TER scores in all the test sets. The automatic MT scores for two test sets, using manual sentence segmentation, are reported in Table 3.92.

		WER	MT results on ASR output and references	
			BLEU	TER
Dev07	(asr output)	10.8	30.1	52.2
	(references)	-	30.6	52.6
Eval06	(asr output)	23.0	21.7	62.6
	(references)	-	21.8	61.3

Table 3.92: Effect of ASR performance on follow-up MT performance.

Ongoing work that has not yet been integrated in the system includes improved pronunciation generation and modeling, unvowelized acoustic models as an alternate system, factored LMs, partial diacritized vocabulary for LM and system optimization for MT purposes.

3.7 Integration of Speech Recognition and Translation

3.7.1. Introduction

Author: Lidia Mangu

Spoken language translation systems have two parts, a Speech-to-Text (STT) system which takes the speech signal and produces a transcript, and a machine translation system, which translates the transcript in another language. The two components are often developed independently of each other. In order to build speech translation systems, it is not enough to just treat the two components as two black boxes and just send the output of the STT system to the MT system. A tighter integration is needed for improving the overall accuracy of the translation. The STT systems are tuned to minimize word error rate (WER) and produce transcripts using automatic segmentation based on voice activity detection. These transcripts do not include punctuation; they contain recognition errors, and disfluencies, especially in the case of spontaneous speech. In comparison, the MT systems are mostly trained on text input containing well formed sentences with punctuations, so this is the type of data they are prepared to deal with at test time.

Each of the three sections in this section attempts to alleviate this mismatch in a different dimension.

The section by Paulik *et al.* (2008) investigates methods for punctuation recovery, disfluency removal and confusion networks (CNs) translation. The classifier used for restoring periods and commas in the STT transcript is a decision tree built with word duration, prosody, silence duration and language model scores as features. The section uses a noisy-channel approach to remove three types of disfluencies: fillers, repetitions and corrections, and shows improved translation performance. This section also deals with improving the MT robustness against STT errors. Instead of using the single best hypothesis from the STT system, the MT system receives as input a confusion network containing a number of word alternates at each position in time. The authors implemented a phrase-based MT decoder of CNs within their beam-search decoder, and showed improvements in BLEU scores compared to using the single best STT hypothesis.

The section by Saon (IBM) presents a method for dealing with STT errors which uses lattices instead of confusion networks as input to the MT system. He gives a recipe for lattice translation which consists in two steps: a monotone lattice decoding which provides the best foreign path, which is then decoded using standard sentence-based MT, with reordering. Improvements in TER are reported on a number of test sets.

The section by Matsoukas *et al.* (BBN,CU,Limsi) addresses the issue of optimizing STT systems for WER, which is not always the best in terms of MT performance. System combination is a standard component of state-of-the-art STT systems; ROVER, CNC and cross-adaptation being the widely used techniques due to the fact that they significantly reduce WER. The section investigates the impact of different system combination strategies on the MT output. The authors describe a method for optimizing the ROVER parameters so as to directly maximize the MT accuracy.

3.7.2. Improving Machine Translation of Spoken Language

Authors: Matthias Paulik, Ian Lane, and Tanja Schultz

3.7.2.1 Introduction

Most machine translation training data consists of written text corpora with well-formed sentences and correct punctuation. ASR output, on the other hand, consists of non-sentence like ‘chunks’ (utterances) of non-punctuated hypotheses. Recognition errors and speech disfluencies like fillers, repetitions and corrections further increase the mismatch between ASR output and MT training data. This severe data mismatch leads to suboptimal translation performance when applying MT directly to ASR output in the context of SLT. One important step to reduce the mismatch is to re-segment ASR hypotheses into sentence-like units before performing translation. Punctuation recovery in ASR output, punctuation removal in MT training data or a mixed approach can further reduce the data mismatch. In addition to improving translation accuracy, sentence segmentation and punctuation recovery can significantly increase the readability of SLT system output.

3.7.2.2 Sentence Segmentation and Punctuation Recovery

In this section, we report results for sentence segmentation and punctuation recovery of periods and commas in the context of SLT. We measure the success of our approaches in terms of end-to-end translation performance using the BLEU metric on human translation references that include correct punctuation.

3.7.2.2.1 Comma Recovery via Modified Phrase Tables

Punctuation recovery for SLT can either be performed before, after or implicitly during translation. Punctuation recovery during translation can be realized by removing punctuation from the source side of the parallel MT training data while retaining punctuation on the target side. However, (Al-Onaizan and Mangu 2007) point out that this will likely degrade word alignment accuracy as it may not be possible to correctly align target punctuation. Instead, they propose to remove punctuation from the source

phrases in the phrase table of a phrase-based statistical MT system, while retaining punctuation in the target phrases. Punctuation recovery in ASR output can rely on acoustic features (prosody, pause duration, etc.) that are extracted from the speech signal. We found that such features are especially of value for the recovery of periods. However, recovering punctuation in the source can degrade overall translation performance. Errors in punctuation recovery can split up long source phrase matches into short phrases and these errors are not recoverable. Punctuation recovery during or after translation does not suffer from this problem, as non-punctuated source phrases guarantee a maximal match. Further, these approaches are more strongly influenced by the target language model, which possibly results in a more 'natural' punctuation of the translation output. We found that these considerations are especially important in the context of comma recovery.

In this work we explicitly distinguish between two separate tasks; sentence segmentation, and punctuation recovery. A sentence segment constitutes the unit presented to MT; MT processes each unit independently, one after another. A sentence segment may include one or more punctuation marks (period, comma) - or none at all. By explicitly separating sentence segmentation from punctuation restoration, it is possible to fully explore punctuation recovery before, after or during translation.

First, we examined the effect of performing punctuation recovery before and/or during translation on the TC-STAR English-to-Spanish SLT task (Paulik 2008; TC-STAR 2004). We applied a punctuation recovery approach that inserts periods and commas into English ASR output based on local language model (LM) context and pause duration between words. A period is explicitly inserted for pause durations $p > 0.7$ seconds. For $0.03s < p \leq 0.7s$, the insertion of a period or a comma is determined by LM probability. After punctuation recovery, the ASR output was resegmented into translation units at each period. We then compared English-to-Spanish SLT performance when: 1) retaining automatically recovered periods and commas in the ASR output, 2) removing all punctuation marks and applying implicit recovery of periods and commas during translation, and 3) retaining periods in the ASR output and recovering commas during translation. For 2) and 3) we initially trained phrase tables with punctuation and then removed punctuation from the source side as described by Al-Onaizan and Mangu (2007). Experiments were conducted on English ASR hypotheses as provided by TC-STAR during the 2006 and 2007 SLT evaluation tasks. We compared the performance of the three methods described above. The 2006 data served as our development set and for the test set we used the 2007 data. Both sets have a word error rate (WER) of 6.9%.

full stop	comma	dev set	test set
<i>source</i>	<i>Source</i>	38.70	36.57
<i>phrase table</i>	<i>Phrase table</i>	39.23	38.29
<i>source</i>	<i>Phrase table</i>	40.16	39.00

Table 3.93: English-to-Spanish BLEU scores for different punctuation recovery schemes.

Compared to a human reference segmentation of the English source, the described automatic sentence segmentation provided an F-measure of 54.79 on the development and 52.48 on the test set. Table 3.93 shows the achieved SLT performance. Inserting full stops prior to MT, on the *source* side at each segment end and recovering commas

implicitly during translation using a modified *phrase table* obtained the best translation performance.

For the GALE SLT tasks we used method 3) which obtained the best performance on the TC-STAR English-to-Spanish task. For Chinese-to-English and Arabic-to-English the method yielded gains of 2.2 and 1.8 BLEU points, respectively, when using audio segmentation and a period at each segment end.¹⁷

3.7.2.2.2. Decision Tree-Based Sentence Segmentation

To train Arabic and Chinese sentence segmentation systems for the GALE SLT tasks, we used four shows from the Arabic BNAD05 data set, six shows from the GALE '06 Chinese development set and half of the GALE '07 Chinese development set. Statistics for our Chinese and Arabic development and test sets are shown in Table 3.94. These sets were extracted from shadow data included in the ROSETTA team GALE 2007 ASR evaluation output. The shadow data was comprised of shows from the GALE 2007 development set and the GALE 2006 evaluation set.¹⁸

	Chinese		Arabic	
	dev	test	Dev	test
GALE source	07dev	06eval	07dev	06eval
Running chars/words	23K	8K	8K	9K
CER/WER	10.5%	17.1%	12.1%	21.7%

Table 3.94: Data statistics for GALE Chinese and Arabic development and test sets.

Segmentation	Punctuation rec.		Chinese-to-English		Arabic-to-English	
	<i>full stop</i>	<i>Comma</i>	dev	test	dev	Test
	audio	<i>yes</i>	<i>no</i>	8.31	8.72	19.46
Audio	<i>yes</i>	<i>yes</i>	8.77	10.09	21.25	15.30
decision tree	<i>yes</i>	<i>yes</i>	8.92	10.73	21.40	15.53
Human	<i>yes</i>	<i>yes</i>	9.38	10.97	21.80	15.96

Table 3.95: Chinese-to-English and Arabic-to-English BLEU scores using different segmentation schemes.

Segmentation	Chinese		Arabic	
	Dev set	Test set	Dev set	Test set
<i>Audio</i>	30.75	31.59	33.89	37.38
<i>Decision tree</i>	59.16	53.38	40.97	43.41

Table 3.96: F-measure of ASR audio segmentation and automatic sentence segmentation.

The last rows in Table 3.95 show the SLT performance achieved when automatically aligning ASR hypotheses to the human reference transcriptions. Human sentence segmentation consistently outperforms audio segmentation. Therefore, it is desirable to automatically introduce human like sentence segmentation on ASR output before MT.

¹⁷ We observed that inserting periods at each segment end leads to improved translation performance, even for speech/non-speech audio segmentation. This concurs with the results presented by Yaser (2007).

¹⁸ In all cases separate training, development and evaluation data was used to evaluate decision tree performance.

Table 3.96 compares the F-measures of the Chinese and Arabic automatic sentence segmentation with the F-measures of the respective audio segmentation. The respective gains in SLT performance for using automatic sentence segmentation are listed in Table 3.95. By combining comma recovery during translation via modified phrase tables with decision tree based automatic sentence segmentation and marking each segment end with a period, we significantly improved SLT performance on both SLT tasks within GALE. Chinese-to-English translation performance improved from 8.72 to 10.73 BLEU points and Arabic-to-English performance improved from 13.51 to 15.53 BLEU points.

3.7.2.2.3. Intra-Sentence Segmentation

In the previous section we showed that human sentence segmentation of ASR output consistently outperforms audio segmentation in the context of SLT. However, the question arises if another form of segmenting input to MT may not outperform human sentence segmentation in terms of translation performance. Rao *et al.* (2007a) reported that introducing intra-sentence segmentation on top of human sentence segmentation can obtain an improvement in BLEU score of up to 6% for Arabic broadcast news data and up to 11% for Arabic broadcast conversation data. This additional intra-sentence segmentation was tuned towards BLEU metric by manipulating the average segment length.

We conducted experiments to introduce such an additional intra-sentence segmentation based on the average segment length (Paulik 2008). However, using the English-to-Spanish SLT system described in 3.7, we were not able to reproduce the results reported by Rao *et al.* (2007a). Instead of finding an optimal value for an average segment length governing the introduction of intra-sentence segments, we rather observed a consistent and ungraceful degradation in BLEU metric when constraining the average segment length to successively smaller values. A possible explanation for these inconsistent results is provided by Rao *et al.* (2007a), no separate MT system parameter optimization was performed after switching between different segmentations, nor was the optimal segment length first estimated on a development set and then tested on a separate evaluation set. In future works we intend to investigate this problem, namely, joint optimization of MT and sentence segmentation, further.

3.7.2.3 Speech Disfluency Removal

Speech disfluencies are generally defined by Tree (1995) as ‘phenomena that interrupt the flow of speech and do not add propositional content to an utterance’. In other words, disfluencies are those parts of spontaneous speech, which when removed, yield the sentence as it was originally intended. Disfluencies can hurt translation quality in two different ways. Firstly, disfluencies such as fillers and repetitions make utterances longer without adding semantic information. On the other hand, corrections add spurious content and, therefore, produce inaccurate translations. Secondly, since SLT systems are trained on large amounts of text data that have well-formed sentences, disfluencies cause a mismatch between training and evaluation data.

We investigated the effect of disfluencies on SLT and presented results for automatic disfluency removal (DFR) that is applied as a preprocessing step prior to translation (Rao

et al. 2007b). Although several different types of disfluencies have been defined (Shriberg 1994), we handle only three: fillers, repetitions and corrections.

	shows	sentences	words	disfl.	precision	recall	F-measure
Training	19	6370	46.3K	6.5%			
BCAD05	5	691	10.6K	3.48%	88.77	90.21	89.48
GALE06	5	256	4.5K	5.86%	95.59	57.57	71.86

Table 3.97: Training and evaluation data statistics for automatic DFR and performance of automatic DFR on manual transcriptions in terms of precision and recall.

The disfluency removal (DFR) system (Honal 2005) uses a noisy-channel approach where DFR is modeled as the translation of disfluent speech to clean speech. Translation during DFR involves only deciding which source words are to be deleted. No word insertions or reorderings are permitted. Five translation models are used capturing features such as; 1) position in sentence, 2) position within the current disfluent region, 3) words in the context of a disfluency, 4) distance from a fragment, and 5) information about previous deletions. These models are combined log-linearly using weights learned via gradient descent. Gradient descent maximizes disfluency removal accuracy on a development set. The five translation models were trained on Arabic broadcast news transcriptions (19 Al Jazeera shows), which had been annotated for disfluencies by a native Arabic speaker. Two additional shows were used as the development set to estimate model weights. Table 3.97 shows the corpus characteristics for the training and evaluation data.

First, we tested the DFR system on two distinct data sets; *BCAD05* and *GALE06* (compare Table 3.97). For a first assessment of the DFR system, we used the manual transcriptions of both evaluation sets as input to the DFR system. The performance of DFR in terms of precision and recall is shown in Table 3.97. A disfluency is correctly deleted if every constituent word is deleted and no other surrounding words are affected. While *BCAD05* consists of Al Jazeera shows, *GALE06* data is comprised of shows from TV channels that are not represented in the training data. We believe that this explains the relatively lower recall on *GALE06* data.

Next, we evaluated the effectiveness of DFR on translation quality. DFR was applied as a preprocessing step to the manual transcriptions before performing translation with the CMU GALE-P2 Arabic-English SMT system. The impact of DFR on translation quality in terms of BLEU metric, is shown in Table 3.98. Both manual removal of disfluencies, as well as automatic DFR improved translation performance. Although the bitext used to train the SMT models included transcripts and translations of broadcast conversations, which contain disfluencies, this is only a fraction of the training corpora. The majority consists of written text that contains no disfluencies, thus DFR reduces the mismatch with the models applied during translation and improves translation accuracy. It is noted that both data sets include a large number of sentences that do not contain disfluencies. In particular, 48% of sentences in *GALE06* data and 76% of sentences in *BCAD05* contained no disfluencies. A more detailed analysis revealed that, due to the high precision of the automatic DFR system, no deterioration in translation performance was observed for sentences without disfluencies. Of the three types of disfluencies

considered in this work, fillers were the most common. Almost 20% of *BCAD05* sentences contain fillers while less than 5% of the sentences contain repetitions or corrections.

DFR	none	automatic	manual
BCAD05	11.13	11.55	11.68
GALE06	12.16	12.36	12.30

Table 3.98: DFR on manual transcriptions and its effect on translation performance (BLEU).

Thus far, we presented results for DFR on manual transcriptions. However, in SLT we apply MT on error-prone ASR output. ASR systems use acoustic models that detect non-verbal speech events such as hesitations and other non-human noise events. These acoustic models are explicitly used to recognize fillers. In order to show the effect of DFR in general, we compare two types of first best ASR output; 1) unmodified first best (UFB) retains filler words hypothesized by the ASR decoder, and 2) modified first best (MFB) removes these filler words. Results for applying automatic DFR to modified and unmodified, first best output, are listed in Table 3.99. A gain of 0.89 BLEU is obtained by merely removing fillers hypothesized by ASR. However, automatic DFR mostly fails to yield any gains, even when applied to the unmodified first best.

	data set	no DFR	automatic DFR
UFB	BCAD05	10.45	10.58
	GALE06	9.86	9.76
MFB	BCAD05	11.54	11.47
	GALE06	10.22	10.18

Table 3.99: Translation results in BLEU using ASR first best output.

The ASR system introduces errors in transcribed output which can be classified as deletion, substitution or insertion type errors. Each of these errors makes DFR challenging; they either distort the lexical context of a DF or altering the DF itself. For example, during ASR decoding the LM gives a low probability to word repetitions and fillers and hence is likely to cause either a substitution or deletion error. In order to quantify the effects of recognition errors, we aligned the ASR first best unmodified output to the DF annotated transcripts and observed what exactly happens to disfluent regions after ASR.

	data set	Del %	Sub %	Ret %
c & r	BCAD05	33.33	24.50	42.10
	GALE06	40.60	50.00	9.40
fillers	BCAD05	39.80	40.40	19.80
	GALE06	37.73	27.35	34.90

Table 3.100: Recognition errors on corrections and repetitions (c & r) and fillers.

Table 3.100 shows recognition errors pertaining to different disfluencies types on *BCAD05* and *GALE06* data. For both data sets, the majority of filler words were either substituted or deleted although the exact proportion depends on specific set. In the case of

repetitions and corrections, a large majority are substituted or deleted. For GALE06 data, fewer than 10% (24 out of 320) were retained. Thus, when presented with ASR output containing word errors involving both disfluent words and their non-disfluent context, disfluency models trained on manually transcribed data perform poorly on the DF removal task.

3.7.2.4 Improving Machine Translation Robustness for SLT

In addition to the problem areas described in the previous sections, one major challenge in spoken language translation is to realize MT that is robust against speech recognition errors. Traditionally, automatic translation of spoken language is implemented by cascading the single best hypothesis from ASR to MT. Translation is then performed as if the input was error free. As ASR output may contain transcription errors and a more accurate recognition hypothesis may be contained in competing hypotheses, this interface should be extended to enable MT to consider multiple ASR hypotheses.

Previous works have investigated numerous approaches to enrich the interface between ASR and MT, these include; translation of N -best lists (Zhang 2004), translation of ASR word-graphs (Matusov 2005), translation of confusion networks (Bertoldi 2005), and joint search of ASR and MT (Amengual 1997). Of these approaches, translation of confusion networks has been shown to be an efficient integration strategy (Bertoldi 2008) which readily scales to large tasks, such as those encountered in the GALE project.

3.7.2.4.1. Statistical Machine Translation of Confusion Networks

In statistical MT (Brown 1993) a foreign language sentence f is translated into another language e by searching for the hypothesis with maximum likelihood \hat{e} where:

$$\hat{e} = \operatorname{argmax}_e P(e|f) \quad (3.60)$$

In the maximum entropy framework the conditional probability $P(e|f)$ can be realized as a log-linear combination of feature functions $h_m(e, f)$ and scaling factors $(\lambda_1, \lambda_2, \dots, \lambda_M)$ giving:

$$\hat{e} = \operatorname{argmax}_e \frac{\exp(\sum_{m=1}^M \lambda_m h_m(e, f))}{\sum_e \exp(\sum_{m=1}^M \lambda_m h_m(e, f))} \quad (3.61)$$

During search the denominator is dropped since it is constant for a given the input f . When translating text input, feature functions including: target language model likelihood, multiple translation model scores (based on lexical translation probabilities), number of words and phrases in the hypothesis, and a distance based reordering penalty, are applied. Scaling factors $(\lambda_1, \lambda_2, \dots, \lambda_M)$ are optimized via MERT (minimum error rate training) (Och 2003) for a specific translation metric such as BLEU.

For spoken language translation the SMT formalism can be extended to consider the uncertainty inherent in speech recognition output. Given a set of transcription hypotheses $\mathcal{F}(\mathbf{o})$ for an observed foreign language utterance \mathbf{o} :

$$\hat{\mathbf{e}} = \operatorname{argmax}_e \sum_{\mathbf{f} \in \mathcal{F}(\mathbf{o})} P(\mathbf{e}, \mathbf{f} | \mathbf{o}) = \operatorname{argmax}_e \exp(\sum_{m=1}^M \lambda_m h_m(\mathbf{e}, \mathbf{f}, \mathbf{o})) \quad (3.62)$$

The likelihoods of ASR hypotheses are incorporated by including additional feature functions for acoustic likelihood $P(\mathbf{o} | \mathbf{f})$, word-sequence probability $P(\mathbf{f})$, or a normalized confidence measure. For translation of confusion networks the ASR posterior probability is used.

A confusion network (CN) (Mangu 2000), is a compact representation of a subset of recognition hypotheses. CNs are weighted directed graphs, where each arc contains a word label, (or a null word, typically notated as ε) and a posterior probability. Each path from the start-node to the end-node goes through all other nodes. Confusion networks are an efficient structure for SMT decoding as word sequences that occur in the same temporal span on multiple paths in a word lattice, appear only once in the CN. This minimizes redundancy during phrase match and significantly reduces the translation search space.

We implemented phrase-based SMT decoding of CNs within our beam search decoder, STTK (Vogel 2003). Decoding is performed in two explicit stages. First, a translation-lattice is build by adding arcs to the input CN for all phrase-pairs in the translation phrase table where the source-side word-sequence matches the word-labels on the CN (ε arcs are skipped when performing this match). The ASR posterior probability of a phrase pair is calculated as the product over the individual arcs it covers on the CN. Beam-search is then performed allowing limited reordering. The final hypothesis is that with maximum score, summed over all feature functions.

3.7.2.4.2. Experimental Evaluation

We evaluated the effectiveness of CN translation on the GALE Chinese-to-English broadcast-news translation task. Details of the experimental setup are shown in Table 3.101. To mitigate the effects of audio segmentation and punctuation, manual utterance boundaries were used to segment the input audio and punctuation was removed from the SMT training corpora. ASR was performed with our Janus decoder using a development version of our Mandarin STT system. Word lattices were generated via ASR and converted to CNs using an enhanced version of the SRILM toolkit. Phrase-based SMT was then applied to the resulting CNs using monotonic decoding. For each input type: manual transcripts, 1-best ASR input and CNs, MT parameters were optimized (via MERT) on the development set.

ASR Evaluation System	
Segmentation	manually defined boundaries
Training Corpora	AM: 500hours, LM: 100M words vocabulary: 107K
Decoding	1-pass speaker-independent AM
SMT Evaluation System	
Phrase Extraction	Alignment: IBM4 (GIZA) Max. Phrase Length: 7
Training Corpora	TM: 100M words LM: 600M words
Development Set	RT-04, 4 shows, 292 utterances
Evaluation Set	RT-04, 3 shows, 652 utterances

Table 3.101: CN translation experimental setup.

First, we evaluated the translation accuracy when MT was applied to manual transcripts and 1-best ASR output. On manual transcripts a BLEU score of 15.50 (single reference, non-punctuated) was obtained on the evaluation set. Applying MT to the 1-best ASR output (which had a WER of 37.2% and a CER of 24.8%) resulted in a BLEU score of 12.76. This large degradation in BLEU is a direct result of recognition errors in the input transcripts.

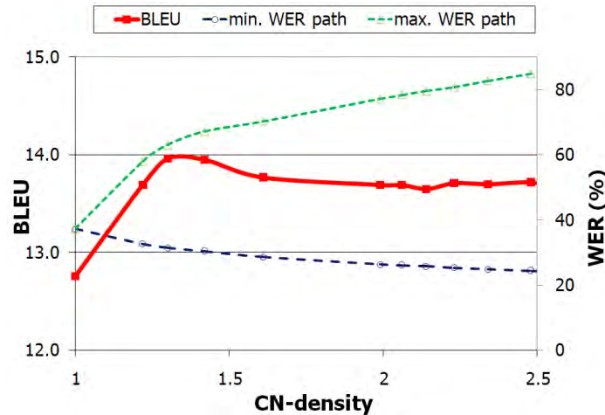


Figure 3.38: Translation accuracy (BLEU) of CN translation compared to average CN density. Dashed lines indicate the average minimum and maximum word error paths through the set of CNs.

Next, we investigated the role of CN density on translation performance. Here, density is defined as the number of non-epsilon edges over the number of nodes in a CN. CNs of varying densities were generated from the original ASR word lattices by pruning lattices with a posterior threshold before generating CNs. CN translation was performed using parameters optimized via MERT for that specific set.

Figure 3.38 shows the BLEU scores obtained when our phrase-based SMT system was applied to CN input. For all densities CN translation obtained higher BLEU compared to using the 1-best ASR input. As the average density of CNs increased, translation performance also initially improved, but then stagnated at higher densities. As density increases it is more likely that a CN contains the correct transcript as a hypothesis path. However, the number of incorrect competing hypotheses also increases exponentially.

Thus, increased CN density does not guarantee improved translation performance. In this work, the best performance was obtained with an average CN density of 1.4, resulting in a BLEU score of 13.96. The improvement in BLEU level compared to the 1-best ASR case (12.76) is significant at the 99% confidence level.

Finally, we analyzed the effectiveness of CN translation for specific ranges of recognition accuracy. Utterances were clustered based on the WER of their 1-best ASR hypothesis. Translation performance was then evaluated on each set. BLEU scores for each input type: manual transcripts, 1-best ASR input and CNs, are shown in Figure 3.39.

It was observed that CN translation was most effective for utterances where the 1-best ASR-WER was between 10% and 50%. For utterances where the WER was greater than 50% CN translation did not significantly outperform the 1-best ASR system. For these utterances the large number of speech recognition errors may make it difficult for translation to effectively leverage surrounding context. Similar results were obtained when reordering was applied during SMT decoding rather than monotonic decoding. Using a reordering window of 4 (RW=4) the performance of all systems improved, and CN translation obtained a BLEU-score of 14.28 on the evaluation set, this compares to 12.90 for the 1-best ASR case. Again, the difference is significant at the 99% level.

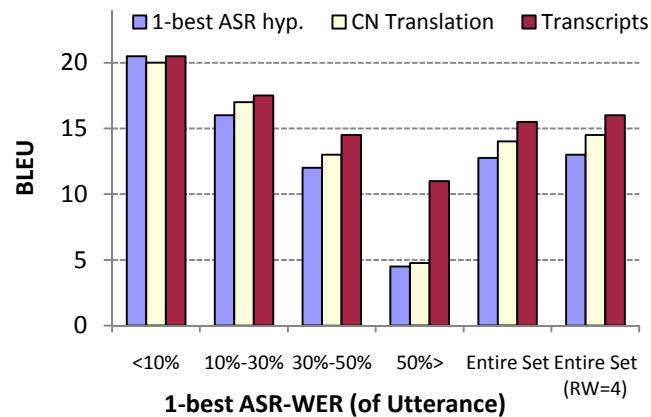


Figure 3.39: Effectiveness of CN translation for different word error rates. Entire Set indicates the performance over the entire evaluation set. RW=4 indicates that a reordering window of 4 was used during SMT decoding. All other results use monotonic decoding.

3.7.2.5 Conclusions

We investigated a number of approaches to overcome the inherent mismatch between automatic speech recognition (ASR) and machine translation (MT) components in spoken language translation (SLT). Specifically, we investigated three problem areas that degrade performance in end-to-end SLT: 1) sentence segmentation and punctuation recovery, 2) speech disfluencies, and 3) MT robustness against ASR errors.

For 1), we proposed a decision tree-based approach that incorporates both, linguistic and acoustic features to automatically segment ASR output. This approach improved

BLEU score by up to 10% compared to using boundaries defined by pause length and speaker change alone. We also determined that generation of end-of-sentence punctuation before translation and generation of intra-sentence punctuation during translation was the most effective scheme to recover punctuation. For 2), we showed that speech disfluencies degrade SLT quality and introduced an Automatic Disfluency Removal scheme which is applied as a preprocessing step prior to translation. And finally, for 3), we implemented phrase-based SMT of confusion networks which improved translation quality (BLEU) compared to using the single best ASR hypothesis. We also found that confusion network translation was most effective for segments with moderate recognitions errors (WER<50%).

3.7.3. Lattice-Based Viterbi Decoding Techniques for Speech Translation

Author: George Saon

3.7.3.1 Introduction

Current research in speech translation focuses on augmenting the interface between ASR and MT to more than 1-best hypotheses, the idea being that the MT component should have the freedom to select paths whose translations are more likely. We distinguish between the *cascaded approach* to speech translation, where the system is comprised of a series of individual engines that are applied in sequence, and the *integrated approach*, where the SMT component accepts multiple ASR hypotheses as input. In increasing order of generality, these hypotheses can be presented in various forms: *n*-best lists, confusion networks (Shen *et al.* 2006) and arbitrary lattices (Mathias and Byrne 2006; Zhou *et al.* 2007; Dyer *et al.* 2008; Matusov *et al.* 2008). The maximum degree of generality (or the tightest coupling) is attained by performing a joint ASR and MT search, which has only been possible, so far, for limited domain tasks (Matusov *et al.* 2005).

There is no guarantee, at the outset, that lattice-based SMT is bound to improve translation performance. This is because ASR lattices contain paths, which are worse than the 1-best in terms of word error rate in addition to the paths, which are better. This is an often overlooked aspect of lattice processing and researchers tend to focus only on the oracle word error rate (i.e., the error rate of the path of lowest WER). It can happen, however, that a weak MT component prefers paths which result in worse overall performance. Therefore, care has to be taken on how to combine the ASR and MT scores when doing lattice decoding.

While lattice-based SMT decoding has been studied previously already (Mathias and Byrne 2006; Zhou *et al.* 2007; Matusov *et al.* 2005; Dyer *et al.* 2008; Matusov *et al.* 2008), the work presented here exhibits differences along several axes. Firstly, compared to the prior art, we discuss how to deal with exhaustive word reordering on a lattice during decoding. To the best of our knowledge, this is only explored by Dyer *et al.* (2008) and, recently, by Matusov *et al.* (2008). The work described by Dyer *et al.* (2008)

is in the context of non-Viterbi decoding for non-ASR lattices such as Chinese word segmentation lattices. Matusov *et al.* (2008) propose a shortcut to deal with word reordering: word links in the lattice are augmented with position (or slot) information from a confusion network processing of that lattice and distortions are computed relative to those positions. In contrast, we are advocating the use of path-dependent distortions, which has the advantage of being theoretically more sound at the expense of increased computational complexity.

Secondly, we propose a decoding strategy which breaks the search into two independent sub-problems: (1) lattice-based monotone search with tracing of the best foreign path, and (2) sentence-based decoding with reordering of that path. Thirdly, we apply lattice MT decoding to a truly large scale speech translation task.

3.7.3.2 Problem Formulation

We are given $\mathbf{x} = x_1 \dots, x_T, x_t \in \mathbb{R}^d$, a sequence of acoustic feature vectors corresponding to a foreign speech utterance which is to be translated into a target string $\mathbf{e} = e_1 \dots, e_I, e_i \in E$. The goal of speech translation is to find the target string $\hat{\mathbf{e}}$ with the highest posterior probability given the acoustics \mathbf{x} :

$$\hat{\mathbf{e}} = \operatorname{argmax}_{\mathbf{e} \in E^*} P(\mathbf{e}|\mathbf{x}) \quad (3.63)$$

The difference with text translation is that the foreign sequence is not given, instead we have to sum up over all possible realizations $\mathbf{f} = f_1 \dots, f_J, f_j \in F$ as suggested by Ney (1999):

$$P(\mathbf{e}|\mathbf{x}) = \sum_{\mathbf{f} \in F^*} P(\mathbf{e}, \mathbf{f}|\mathbf{x}) \quad (3.64)$$

By applying Bayes rule to the term on the right side of Equation (3.64), we get:

$$P(\mathbf{e}, \mathbf{f}|\mathbf{x}) = \frac{P(\mathbf{x}|\mathbf{e}, \mathbf{f})P(\mathbf{e}, \mathbf{f})}{P(\mathbf{x})} = \frac{P(\mathbf{x}|\mathbf{f})P(\mathbf{e}, \mathbf{f})}{P(\mathbf{x})} \quad (3.65)$$

where in the last Equation (3.65) we made the (reasonable) assumption that the acoustic sequence does not depend on the target word sequence when conditioned on the foreign word sequence.

By combining (3.63), (3.64) and (3.65) and by taking into consideration that $P(\mathbf{x})$ is constant with respect to the max operation, we obtain:

$$\hat{\mathbf{e}} = \operatorname{argmax}_{\mathbf{e} \in E^*} \sum_{\mathbf{f} \in F^*} P(\mathbf{x}|\mathbf{f}) P(\mathbf{e}, \mathbf{f}) = \operatorname{argmax}_{\mathbf{e} \in E^*} \sum_{\mathbf{f} \in F^*} P(\mathbf{x}|\mathbf{f}) P(\mathbf{f}|\mathbf{e}) P(\mathbf{e}) \quad (3.66)$$

with $P(\mathbf{x}|\mathbf{f})$ being the likelihood of the acoustic feature vectors given the foreign word sequence, $P(\mathbf{f}|\mathbf{e})$ the likelihood of the foreign word sequence given the target words and $P(\mathbf{e})$, the prior probability of the target word sequence (given by the English language model). In Equation (3.66) we have applied the source-channel model for machine translation (Brown *et al.* 1993) to separate the two knowledge sources: the LM term which controls the well-formedness of the target word sequences and the translation model. Alternatively, it is also common, especially when modeling the translation

problem with WFSTs, to reason directly in terms of joint probabilities as do Kanthak *et al.* (2005) and Zhou *et al.* (2007).

3.7.3.2.1. Lattice-Based Decoding

ASR lattices are word graphs, which compactly encode the word sequences with the highest likelihood given the acoustics. An example of a word lattice is shown in Figure 3.40.

Lattice-based decoding simply means that we restrict the summation in Equation (3.66) only to the paths occurring in a lattice, that is:

$$\hat{e} = \operatorname{argmax}_{e \in E^*} \sum_{f \in L(x)} P(x|f)P(f|e) P(e) \tag{3.67}$$

where $L(x)$ denotes the set of paths that occur in the ASR lattice corresponding to the acoustics x .

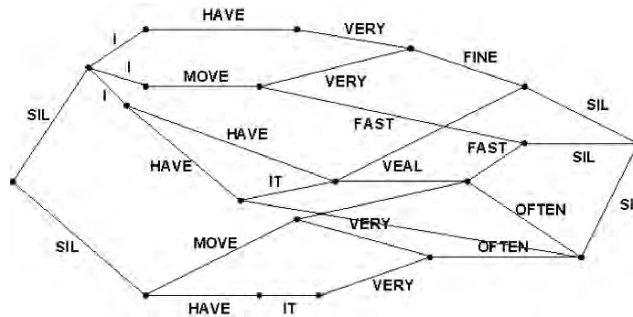


Figure 3.40: Example of a word lattice (SIL stands for silence)

3.7.3.2.2. Phrase-Based SMT

In phrase-based translation (Koehn *et al.* 2003), we segment the foreign input sentence into a sequence of $1 \leq K \leq J$ phrases $\bar{\mathbf{f}} = \bar{f}_1 \dots \bar{f}_K$. We assume a uniform distribution over all possible segmentations. Each foreign phrase \bar{f}_k in $\bar{\mathbf{f}}$ is translated into a target phrase \bar{e}_k with probability $\phi(\bar{f}_k|\bar{e}_k)$. The conditional probability $P(\mathbf{f}|\mathbf{e})$ can be expressed as:

$$P(\mathbf{f}|\mathbf{e}) = \sum_{\bar{\mathbf{f}}, \bar{\mathbf{e}}} P(\bar{\mathbf{f}}|\bar{\mathbf{e}}) = \sum_{\bar{\mathbf{f}}, \bar{\mathbf{e}}} \prod_{k=1}^K \phi(\bar{f}_k|\bar{e}_k) \tag{3.68}$$

where the summation is performed over all the segmentations which are consistent with \mathbf{f} and \mathbf{e} .

3.7.3.2.3. Source Word Reordering

Let $\sigma: \{1, \dots, J\} \rightarrow \{1, \dots, J\}, \sigma \in S_J$, be a permutation of the set $\{1, \dots, J\}$ with S_J being the symmetric group of order J . We define the permuted foreign sequence as $\mathbf{f}_\sigma := f_{\sigma(1)} \dots f_{\sigma(J)}$.

The translation model can be expressed by marginalizing over the permutations and by using the chain rule of conditional probabilities as follows:

$$P(\mathbf{e}, \mathbf{f}) = \sum_{\sigma \in \mathcal{S}_J} P(\mathbf{e}, \mathbf{f}, \sigma) = \sum_{\sigma \in \mathcal{S}_J} P(\mathbf{f}|\mathbf{e}, \sigma) P(\mathbf{e}|\sigma) P(\sigma) = \sum_{\sigma \in \mathcal{S}_J} P(\mathbf{f}_\sigma|\mathbf{e}) P(\mathbf{e}) P(\sigma) \quad (3.69)$$

with $P(\sigma)$ being given by the distortion model. The distortion model used in this work is based on the window/skip model introduced by Tillmann and Ney (2003) and operates at the word level. Note that the derivation in Equation (3.69) assumes that the distortion does not depend on the foreign word sequence (i.e., the model is not lexicalized). More powerful distortion schemes such as the inbound-outbound distortion proposed by Al-Onaizan and Papineni (2006) are lexicalized and can be written as:

$$P_f(\sigma) := \prod_{j=1}^{J-1} P(\sigma(j+1) - \sigma(j) | f_{\sigma(j)}) \quad (3.70)$$

which simply means that we model the probability of the length of the jump to the next word conditioned on the identity of the current word.

Putting Equations (3.67) and (3.69) together, we can formulate the lattice-based SMT decoding problem with reordering in the following way:

$$\hat{\mathbf{e}} = \operatorname{argmax}_{\mathbf{e} \in E^*} \sum_{\mathbf{f} \in L(\mathbf{x})} P(\mathbf{x}|\mathbf{f}) \sum_{\sigma \in \mathcal{S}_J} P(\mathbf{f}_\sigma|\mathbf{e}) P(\mathbf{e}) P_f(\sigma) \quad (3.71)$$

3.7.3.2.4. Two-Pass Strategy

Implementing Equation (3.71) directly can be computationally expensive because of the double summation. The idea then is to divide the problem into two subparts in the following manner:

1. Lattice-based *monotone* decoding of best *foreign* path:

$$\hat{\mathbf{f}} = \operatorname{argmax}_{\mathbf{f} \in L(\mathbf{x})} \max_{\mathbf{e} \in E^*} P(\mathbf{x}|\mathbf{f}) P(\mathbf{f}|\mathbf{e}) P(\mathbf{e}) \quad (3.72)$$

2. Sentence-based *non-monotone* decoding of best *English* path:

$$\hat{\mathbf{e}} = \operatorname{argmax}_{\mathbf{e} \in E^*} \sum_{\sigma \in \mathcal{S}_J} P(\hat{\mathbf{f}}_\sigma|\mathbf{e}) P(\mathbf{e}) P_f(\sigma) \quad (3.73)$$

This strategy has the obvious advantage of factoring the search into sub-problems of lower complexity. In the next section, we discuss how to implement Equations (3.71) – (3.73) efficiently with WFSTs.

3.7.3.3 Decoder Description

3.7.3.3.1. WFST Interpretation

Finite state machines offer a convenient formalism to represent, access and manipulate heterogeneous knowledge sources in a uniform way. Knowledge sources can be combined through the composition operation and the resulting automata can be “shrunk” to an optimal size via determinization and minimization and efficiently searched. It is no wonder then that WFSTs have lately become the formalism of choice in both ASR (Saon *et al.* 2005) and SMT (Zhou *et al.* 2007; Mathias and Byrne 2006; Kanthak *et al.* 2005) and we shall make no exception here.

Similarly to (Kanthak *et al.* 2005), Equation (3.71) can be rewritten using finite state terminology as follows:

$$\hat{e} = \text{best-path}(I \circ R \circ M \circ L) \quad (3.74)$$

where the component FSMs represent:

- I : input lattice acceptor. Encodes f with weight $P(x|f)P(f)$. Theoretically, $P(f)$ should not be included, however, it does help with pruning in practice.
- R: reordering transducer. Maps f to f_σ with weight $P_f(\sigma)$.
- M : translation transducer. Maps f_σ to e with weight $P(f_\sigma|e)$.
- L: language model acceptor. Encodes e with weight $P(e)$.

3.7.3.3.2. Decoding Approaches

Computing the entire composition in Equation (3.74) offline is usually intractable for a larger scale task. The alternative is to have a decoder which implements *on-demand* (or *on-the-fly*) composition during the search. This approach has been favored by Kanthak *et al.* (2005) and Zhou *et al.* (2007). Both authors opt for using general FSM decoders. This has the advantage that the same decoder can be used for different FSM configurations and the disadvantage that the decoder is oblivious to special FSMs (e.g., R). We take a different approach and design a specialized decoder which has an efficient implementation for the reordering FSM (but, cannot support arbitrary FSM configurations).

3.7.3.3.3. Search Space Construction

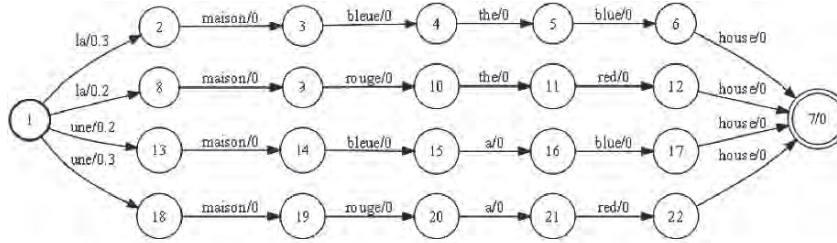
The starting point for creating the M FSM is a bilingual phrase table with scores. Next, we carry out the following steps in sequence which are illustrated in Figure 3.41 for a French-to-English example:

1. Add one path per phrase from the start to the final state with the phrase cost on the first arc.
2. Determinize resulting acceptor.
3. Minimize previous acceptor.
4. Connect final states to start state with epsilon arcs and mark foreign/English arcs.

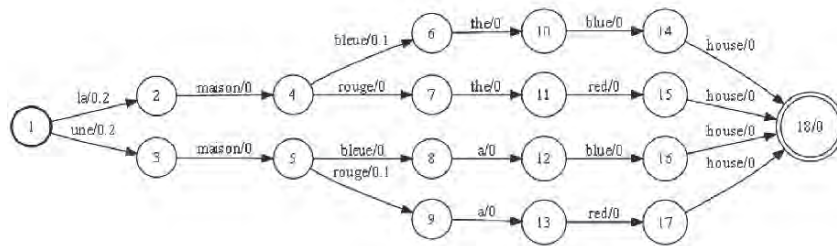
We encoded the phrase table as an acceptor (instead of a transducer) because the determinization and minimization operations are much simpler on acyclic acceptors and we rely on the decoder to differentiate between the foreign and the English arcs. This is done by having two sets of arcs for each state of the phrase-table acceptor: a set of foreign arcs, which can only be traversed when the decoder “consumes” an input word, and a set of English arcs which, when traversed, produce output words.

la maison bleue	the blue house	.3
la maison rouge	the red house	.2
une maison bleue	a blue house	.2
une maison rouge	a red house	.3

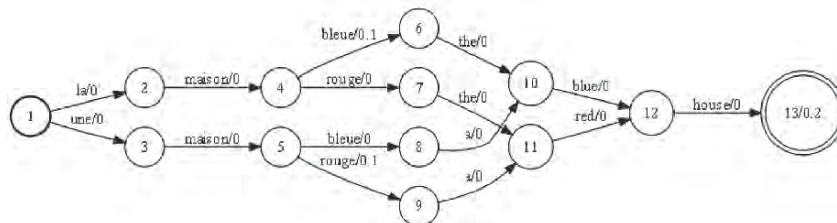
(a)



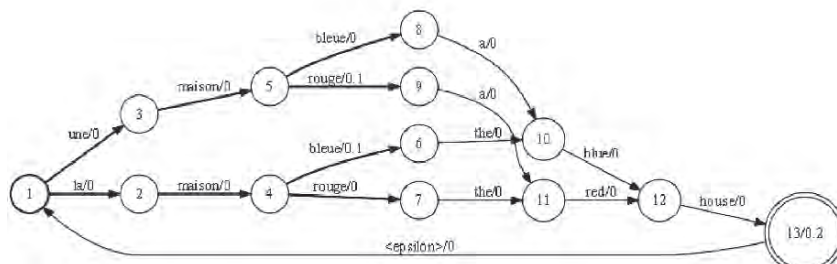
(b)



(c)



(d)



(e)

Figure 3.41: Steps for constructing a phrase-table acceptor: (a) phrase table, (b) acceptor with one path per phrase, (c) determinized acceptor, (d) minimized acceptor and (e) cyclic acceptor.

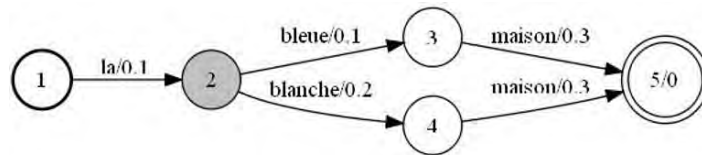
3.7.3.3.4. Viterbi Decoding

<pre> for (l,f,mask,m,n,score) in active tuples rightmost covered := -1 for i from 0 to window size-1 if mask[i] != 0 rightmost covered := i continue endif if i-rightmost covered > skip size+1 break endif successors = LatticeDistantSuccessorArcs(f, i) for larc in successors if larc.word in ForeignSuccessorArcs(m) marc := ForeignSuccessorArc(m, larc.word) score' := score+larc.score*lat weight +marc.score+dist score*dist weight l' := larc.dest state m' := marc.dest state mask' := mask mask'[i] := l' </pre>	<pre> while mask'[0] != 0 f := mask'[0] mask' := ShiftOnePositionLeft(mask') endwhile active successor tuples.append((l',f',mask',m',n,score')) endif endif endfor Prune(active successor tuples) active tuples := active successor tuples for (l,f,mask,m,n,score) in active tuples for marc in EnglishSuccessorArcs(m) score' := score+LangModelScore(n, marc.word) n' := LangModelExtend(n, marc.word) active tuples.append((l,f,mask,m,n',score')) endif endif </pre>
---	---

Figure 3.42: One step of the Viterbi algorithm for lattice decoding with reordering.

The search proceeds in the following way. For each input word f_i , we take simultaneous transitions which are labeled by the same word in pairs of automata: first, we take one pair of transitions labeled by f_i in (I, M) and then, if possible, we take a sequence of transitions labeled by e_{j_1}, \dots, e_{j_k} in (M, L) . Transitions in (I, M) are used to “consume” input words whereas transitions in (M, L) “produce” English words. The search algorithm is described in more detail in Figure 3.42. Note that this requires the “English” section of the phrase-table FSM to be sorted in topological order. The search stops when we reach the final states in all component FSMs. Figure 3.43 shows an example of component FSMs for a French-to-English translation task.

All hypotheses which compete at a given step cover the same number of input words which is commonly referred to as *cardinal synchronous* decoding. These hypotheses are pruned using a combination of beam and histogram (or rank) pruning as explained by Saon *et al.* (2005) for ASR.



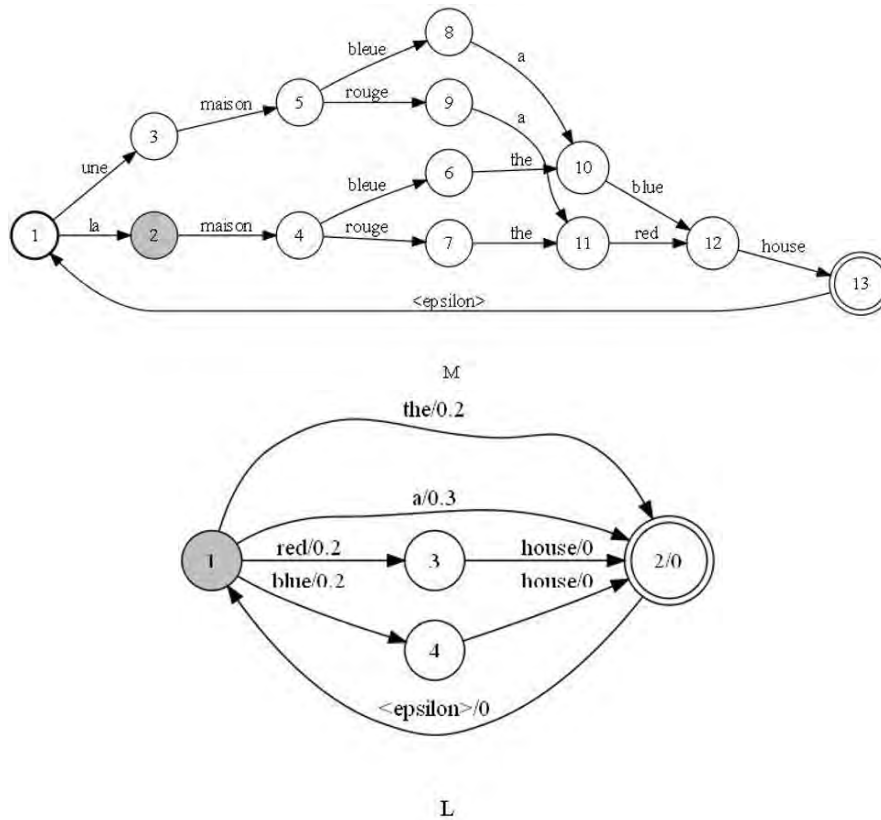


Figure 3.43: Example of component FSMs: ASR lattice L , phrase table FSM M and English language model L . Filled states denote current active states.

3.7.3.3.5. Monotone Decoding

For monotone decoding, we keep track of tuples of active states (l, m, n) , where l is a state in the input lattice, m is a state in the phrase table FSM and n represents a target language model state which is similar to (Zhou *et al.* 2007).

In addition, we keep track of the following component scores, which are combined with independent weights: acoustic and foreign language model scores from the ASR, translation scores, and English LM scores.

3.7.3.3.6. Non-monotone Decoding

In the case of search with reordering, there are a couple of additional complications. First, we have to be able to access distant successors in a lattice from any given node, (e.g., immediate successors have distance 1, their immediate successors have distance 2, and so on). Second, we have to keep track of more information during the search for each hypothesis:

Current state in input lattice (l),
 First state in input lattice from which reordering window applies (f)
 State vector for reordering window (mask) State in translation FSM (m)
 State in LM FSM (n)
 Jump distance from previous word (used for reordering)

The state vector for the reordering window is an extension of the coverage bit vector (Kanthak *et al.* 2005) except that, instead of bits, we have lattice states. It can be represented as a 64 bit integer if we limit ourselves to a maximum window size of 4 and if each lattice has less than 64K nodes. This elegantly solves the problem of path recombination i.e., hypotheses which have identical window values can be merged. Compared to monotone decoding, there is an additional distortion score, which is also combined with an independent weight.

In order to provide a more precise description of lattice decoding with reordering, we give some details in Figure 3.42 of the algorithm for advancing on one input word.

3.7.3.4 Experiments and Results

Name	Genre	Nb. segments	Nb. words
BNAD05	BN	8	15.0K
DEV07	BN+BC	110	19.6K
EVL06bn	BN	24	6.0K
EVL06bc	BC	14	6.7K

Table 3.102: Testset description. Nb. of segments refers to the number of shows or “snippets”. Nb. of words represents the number of *decoded* Arabic words.

The experiments were conducted on a large scale Arabic-to-English broadcast news and broadcast conversations speech translation task, which is part of the GALE DARPA program. We report results on four test sets which vary by collection time and by genre. Table 3.102 summarizes the characteristics of these various test sets.

The speech recognition system used in this work consists in the first two decoding passes of our 2006 GALE evaluation system (which had three passes/models). The first pass search is done with speaker independent acoustic models and the second pass with speaker-adaptive trained (or SAT) models. Both sets of models are trained discriminatively on 135 hours of supervised data and 1800 hours of un-supervised data (i.e., without reference transcripts). The models are unvowelized (or graphemic) in the sense that short Arabic vowels are not explicitly represented. More details about the training of the Arabic models are described by Soltau *et al.* (2007).

BNAD05	DEV07	EVL06bn	EVL06bc
15.7%	17.9%	24.0%	31.9%

Table 3.103: ASR 1-best word error rates.

We generated two sets of lattices which differ in the average link density (ALD) and the degree of pruning: one set with an ALD of 2.5 (pruned with a beam of 0.5), and another one with an ALD of 5 (pruned with a beam of 1.0). The original (unpruned) lattices have an ALD of 450. The ASR decoder that was used to generate the lattices is

described by Saon *et al.* (2005). The 1-best word error rates on the different test sets are summarized in Table 3.103.

The MT phrase tables were trained on a variety of corpora: UN parallel corpus, LDC News and various GALE data releases. The size of the parallel training corpus is 7.3M sentence pairs (roughly 313K pairs after test set sub-sampling). Phrases were extracted using the *inverse projection constraint* described by Och and Ney (2000). We used both HMM and maximum entropy word level alignments. This procedure for extracting phrases is standard for all IBM phrase-based SMT decoders. The English language model is a modified Kneser-Ney smoothed 5-gram LM and has about 800M n -grams. All non-monotone decodings were run with a window width of 4 words and a maximum skip length of 2 words.

	BNAD05	DEV07	EVL06bn	EVL06bc
Monotone decoding				
1bst	64.0%	60.1%	67.6%	70.6%
LS1	63.0%	59.8%	67.4%	69.8%
LS2	63.2%	59.8%	67.8%	69.5%
Non-monotone direct decoding				
1bst	61.7%	58.9%	66.9%	70.3%
LS1	61.5%	59.1%	67.4%	70.8%
LS2	61.6%	59.4%	68.1%	70.4%
Non-monotone two-pass decoding				
1bst	61.7%	58.9%	66.9%	70.3%
LS1	61.2%	58.8%	67.1%	69.9%
LS2	61.3%	59.2%	67.5%	69.3%

Table 3.104: TER results for lattice-based SMT decoding.

In Table 3.104 we report TER results for both monotone and non-monotone decoding for 1-best, lattice set 1 (LS1) and lattice set 2 (LS2). Scoring is case insensitive with a single reference. We also compare direct decoding with the twopass strategy presented in Section 3.7.3.2.4. An analysis of these results show that the largest gains (1 TER point) are obtained for EVL06bc followed by the BNAD05 test-set (1 point monotone, 0.5 with reordering). The gains on the other test sets are inconclusive. A second observation is that the two pass strategy outperforms the direct decoding strategy and degrades less on the larger lattices. We surmise that this is because the two pass strategy copes better with search errors than direct lattice decoding with reordering. We also note that the only test set which improves for the larger lattices is EVL06bc (which, also, has the highest WER/TER).

	Monotone	Reordering
Sentence based	260.3 w/s	13.4 w/s
Lattice based	19.4 w/s	4.1 w/s

Table 3.105: Execution speed in number of translated words per second for the various decoding strategies.

Lastly, the decoding speed of the various approaches (monotone, reordering, sentence based and lattice based) is summarized in Table 3.105 as the number of decoded words per second. All experiments were run on a 2.8GHz AMD Opteron 64-bit processor.

3.7.3.5 Conclusion

We described a Viterbi decoder for speech translation which operates on general ASR lattices. The decoder finds the path of minimum cost through an on-demand composition of several automata. We deal efficiently with word reordering on the lattice by representing the reordering FSM implicitly as a lattice node coverage vector. A twopass decoding strategy is also presented which has the advantage of factoring the search into simpler sub-problems. Future work will extend some of these ideas to joint decoding.

3.7.4. Optimizing Speech-to-Text System Combination for Machine Translation

Authors: Spyros Matsoukas, Bing Zhang, Jeff Ma, Tim Ng, Long Nguyen, Frank Diehl, Mark J. F. Gales, Xunying Andrew Liu, Junho Park, Marcus Tomalin, Phillip C. Woodland, Jean-Luc Gauvain, Lori Lamel, Abdelkhalek Messaoudi, Petr Fousek, and Jun Luo

3.7.4.1 Introduction

System combination has become a standard component of state-of-the-art speech recognition systems. In its simplest form – ROVER (Fiscus 1997) – it combines 1-best speech-to-text (STT) hypotheses from multiple systems to produce a single hypothesis that has lower word error rate (WER) than any of the individual system outputs. Another combination approach, commonly referred to as cross-adaptation, uses one system's STT hypotheses as adaptation data for transforming another system's acoustic and/or language model parameters prior to running a final recognition pass. Other, more elaborate system combination methods, such as consensus network combination (CNC) (Evermann and Woodland 2000), aim at combining lattices from multiple STT systems, and often incorporate cross-adaptation to maximize overall recognition accuracy.

The above techniques have served the speech recognition community well in terms of reducing WER. However, it is not clear whether reducing WER is the right thing to do when the final evaluation criterion is machine translation (MT) accuracy of the STT combination output. A preliminary investigation of STT system combination for MT (Gales *et al.* 2007), revealed that there are indeed certain cases in which a system combination method with higher WER leads to better MT performance – measured in terms of bilingual evaluation understudy (BLEU) or translation edit rate (TER) scores – compared to using a lower WER combination technique.

We continue that investigation, probing deeper into understanding the effects of STT system combination methods on Arabic-English and Chinese-English MT. In addition, we present a novel technique for optimizing the parameters of ROVER STT combination so as to directly maximize MT accuracy.

3.7.4.2 Optimizing STT System Combination

In this section we review the ROVER system combination method and present methods for optimizing the ROVER parameters so as to maximize either STT or MT

accuracy. Later we will contrast these methods to standard ROVER or cross-adaptation, in terms of both STT and MT performance.

3.7.4.2.1. ROVER Background

The first step in ROVER combination is to align all system outputs to each other, forming a consensus network (CN). A CN is a restricted directed graph in which every path from the start node to the end node must go through the same sequence of nodes. Once the nodes are ordered according to a topological sorting of the graph, all arcs coming out of a node must enter the node adjacent to it. All arcs between a pair of adjacent nodes represent the possible distinct word/null choices available at that position in the CN. In addition to the word/null label, each arc also carries information about which system included that word/null at this particular position in its output, along with a confidence score. The confidence score for null arcs is fixed to a constant value across all systems, and it is called the “null confidence” score.

The next step in ROVER is to compute a single score for each arc in the CN. If q is an arc, and $w(q)$ is the word/null label of the arc, then ROVER calculates

$$r(q) = \alpha \frac{\sum_{i=1}^N \lambda_i v_i(q)}{\sum_{i=1}^N \lambda_i} + (1 - \alpha) \frac{\sum_{i=1}^N \lambda_i c_i(q)}{\sum_{i=1}^N \lambda_i} \quad (3.75)$$

where N is the number of systems being combined, λ_i is a system specific weight set to 1 by default, and $v_i(q)$ is a binary variable that is 1 if $w(q)$ is generated by system i and 0 otherwise. When $v_i(q) = 1$, the function $c_i(q)$ returns system i 's confidence score of $w(q)$, and 0 otherwise. When $w(q)$ is NULL, $c_i(q)$ returns the constant null confidence score if system i does not output any word at this position, and 0 if it does. We will refer to the null confidence score as β in the rest of the section.

Thus, the ROVER parameters are α , β , and $\lambda_1, \dots, \lambda_N$, each taking value in the range $[0, 1]$.

Finally, the CN is searched to find the path that has the highest product of ROVER arc scores. Given the topology of the CN, this can be performed simply by finding the outgoing arc q with the highest $r(q)$ value for each node in the CN.

3.7.4.2.2. STT-Based Tuning

The ROVER parameters may be tuned in order to optimize STT performance, such as WER. In standard ROVER, only α and β are used, and can be tuned easily by quantizing the interval $[0, 1]$ into a number of bins, then sampling it for each of the two parameters, running ROVER and evaluating the STT output, and finally choosing the (α, β) setting that provides the best result.

Tuning becomes more difficult when system weights are included, especially as the number of combined systems increases. We have developed a process in which a derivative-free optimization algorithm; in our case, Powell's method (Powell 1964), is employed to find optimal values of the ROVER parameters. In this process, the objective function (e.g., WER) is evaluated several times with varying parameter vector values in order to identify the direction that optimizes the evaluation criterion in the multi-dimensional search space. Evaluation of the objective function can be done in two steps:

first, the 1-best hypothesis for each sentence is extracted from the consensus network using Equation (3.75) to calculate the CN arc scores for ROVER. Then, the 1-best hypotheses of the entire test set are scored to compute the objective function value.¹⁹

3.7.4.2.3. MT-Based Tuning

The ROVER parameters may also be tuned in order to optimize MT performance (TER/BLEU) directly.

One way to do this is to apply a modified ROVER to the CN that can extract N -best STT hypotheses, along with their associated detailed CN arc scores. This is straightforward once the ROVER arc scores have been computed based on Equation (3.75). Then, each of the N -best STT hypotheses can be translated using the baseline MT engine, and the resulting MT hypotheses can be evaluated in terms of TER or BLEU scores. That provides a mapping from the STT ROVER N -best hypotheses to their fitness score in terms of MT, and Powell's method can be used again to optimize the ROVER parameters as follows:

1. Given the current parameter settings, loop over the STT hypotheses in the N -best and compute their total ROVER score as a product of their corresponding updated arc scores.
2. Extract the top scoring hypotheses from the N -best and calculate their fitness in terms of TER or BLEU score, using the mapping mentioned above.
3. Apply Powell's search to find the next setting of the ROVER parameter vector, based on the fitness value.
4. Go to step 1 and iterate until the parameter vector converges.
5. Given the new ROVER parameter settings, go back to the CN and decode it again in order to generate a new N -best of STT hypotheses.
6. Merge new N -best with old and run MT on the newly inserted STT hypotheses to get their TER/BLEU fitness statistics.
7. Go to step 1 and iterate until the parameter vector converges.

The above procedure will eventually lead to ROVER settings that improve MT performance on the tuning set, but it is too costly as it requires translating an N -best of STT hypotheses for each segment in the tuning set. We, therefore, explored an alternative method, in which the MT hypothesis N -best is generated in a single step, by running MT directly on the ROVER CN, using a phrase-based MT system modified as by Bertoldi and Federico (2005) in order to support CN decoding. During this process, additional information is stored in order to reveal for each MT hypothesis its corresponding STT hypothesis in the CN (i.e., the exact sequence of arcs in the CN that the MT hypothesis was generated from). It should be easy to see now that the procedure described previously for optimizing ROVER in terms of MT can be applied here as well. Note, however, that one difference between this N -best method and the previous is that here the N -best may contain multiple MT hypotheses for the *same* STT path in the CN, while in the previous method the mapping between STT and MT hypotheses is one-to-one. This

¹⁹ For Chinese, there is an option to create the consensus networks at the character level. In that case, the 1-best hypotheses need to be re-segmented to words before they are scored in terms of WER.

could cause a premature bias towards the most likely CN path in the early stages of optimization, but we believe that such a problem can be alleviated by generating a deep enough N -best when translating the ROVER CN.

3.7.4.3 Experimental Setup

In this section we describe the setup that was used for all experiments reported. Specifically, we provide details about the development sets, STT systems, and MT systems.

3.7.4.3.1. Development Sets

We ran experiments on the Arabic and Chinese AGILE STT/MT development sets. These sets were created by pooling together episodes from various existing speech test sets (e.g., GALE LDC-released audio development/evaluation sets), and partitioning them into three sets for each language. One set was used for tuning and the other two for validation. Here, we report results on the validation sets, described in more detailed below.

Arabic: The arabic validation sets, at1 and at2, consist of equal parts of news and conversations broadcast from November 2005 through May 2007 by major Arabic-speaking television and radio stations (e.g., Al-Jazeera, Al-Arabiya, Syrian TV). Specifically, at1 is made of 72 episode snippets, containing approximately 5 hours (1527 segments) of speech, while at2 has 134 episode snippets amounting to 4.5 hours (1591 segments) of speech.

Chinese: The Chinese validation sets, ct1 and ct2, consist of equal parts of news and conversations broadcast from February 2001 through May 2007 by major Chinese-speaking television and radio stations (e.g., CCTV, NTDTV, Phoenix, VOA). Specifically, ct1 is made of 69 episode snippets, containing approximately 4 hours (2581 segments) of speech, while ct2 has 177 episode snippets amounting to 3.5 hours (2564 segments) of speech.

It is worth noting that there is only a single English reference translation available for most of the audio snippets in the AGILE tuning and validation sets.

3.7.4.3.2. Speech Recognition

Several STT systems from BBN, University of Cambridge (CU), and LIMSI were combined in various ways in order to produce the results presented in this section. A description of the systems is given below.

Arabic: BBN generated four systems: Phoneme, Grapheme, Word-level and Contextual Morpheme (Ng *et al.* 2009). The language models used in these systems were trained on 1.7 billion words of text, while the acoustic training corpus consisted of 1433 hours of audio data. The 490K vocabulary in the grapheme-based system decoding dictionary consisted of all acoustic training transcript words, plus all words that occurred at least 30 times in the language model training data. About 390K words whose pronunciations could be found in the BBN master dictionary were used to construct the phoneme-based system decoding dictionary. The recognition lexicon for the two

morpheme systems was derived from the word-based lexicon. The word-level morpheme system used a 284K lexicon; a 290K lexicon was used for the contextual system. All acoustic models were gender independent, crossword quinphone state clustered, with a 9-frame PLP long span feature (LSF) (Zhang *et al.* 2004) frontend, and were trained discriminatively based on Minimum Phoneme Frame Error (MPFE) (Zheng and Stolke 2005).

CU employed a multi-pass framework, with four main systems that differed in the frontend and tokenization. Decoding included a common initial lattice generation stage followed by lattice expansion with an interpolated word and class based 4-gram language model. These lattices were then rescored using a number of adapted acoustic models. The language models were trained on about a billion words of text data from a range of acoustic transcriptions and text sources. The word list associated with this language model had 350k words. All acoustic models were state-clustered triphones, built on 1537 hours of acoustic training data. Two forms of feature vectors were used for the acoustic models. The first consisted of 39-dimensional, HLDA-projected, PLP features. The second frontend was a 65-dimensional PLP+MLP, which spliced 26 MLP-based bottleneck features (Grézl *et al.* 2007), generated from wLP inputs (Fousek 2007), to the initial feature vector. A semi-tied transform was applied to the MLP features (Park *et al.* 2009). Both phonetic and graphemic gender dependent minimum phone error trained models were built on each of the frontends. The initial lattice generation was performed using a graphemic system built on PLP features.

LIMSI provided 4 systems, differing in the acoustic feature vector (39 PLP coefficients vs. 78 MLP+PLP) and in the tokenization (words vs. morphemes). Each system used a two pass decoding strategy: first, word (morph) lattices were generated with crossword, gender-dependent SAT, MMIE acoustic models trained on 1200 hours of manually transcribed data; second, the lattices were converted into consensus networks and were decoded with adapted acoustic models, a word- or morph-based 4-gram LM, and pronunciation probabilities (Gauvain *et al.* 2002; Lamel *et al.* 2007). The LM was obtained by interpolating the classical 290k word 4-gram backoff LM and three neural network (NN) LMs. Each 4-gram backoff LM was trained on over 2.6 billion words of texts, and each NN LM on the audio and web transcripts, as well as, a sampling of the other text sources.

Chinese: BBN had six Mandarin STT systems: the first three were updated versions of the systems described by Ng *et al.* (2008), with MPFE acoustic models trained on a 1734-hour audio data set, and 4-gram LM trained on 4.6 billion characters of text. An improved word-compounding mechanism was used for the other three STT systems based on BBN's Chinese parser (Serif). Words were compounded with their nearest sibling words in the parse tree if their n-grams were in the 40K most frequent n-gram list. An LM was estimated on training data that was automatically segmented using the aforementioned compound word list based on the maximum length algorithm (MLA). This LM was then interpolated with the uncompounded LM and the compound-word LM trained on the word segmentation derived from the parse trees.

CU employed four systems, in a decoding structure similar to that described by Sinha *et al.* (2006). It consisted of an initial lattice generation stage followed by lattice rescored with adapted 4-gram language models and acoustic models. The language

model was trained on 2.9 billion words from a total of 27 sources. Two sets of lattices were used in acoustic model rescoring stages. These were expanded from the initial lattices using context dependent language model adaptation (Liu *et al.* 2008). The 1-best hypotheses from the baseline 4-gram lattices were used as the supervision for the LM adaptation. All acoustic models were state-clustered triphones, trained on 1676 hours of acoustic training data using MPE training. Two forms of feature vectors were used for the acoustic models, PLP and PLP+MLP. Two PLP-based systems were built: a gender-dependent system; and a SAT system. For both systems an additional Gaussianization stage was applied to the feature vector. Two PLP+MLP-based systems were also built: a gender-dependent system; and a gender-dependent PLP-Gaussianized system.

LIMSI provided one system with PLP+MLP acoustic frontend (including pitch features), SATMMIE acoustic model training, and classical/NN 4-gram LMs. The acoustic models were trained on 1441 hours of manually transcribed data. A common 56k word recognition vocabulary was used for recognition, which was carried out in two passes (see LIMSI's Arabic STT description above). The NN language models were trained on a subset of the available data including the audio transcripts (17.3M words) and 1.3 billion words from the most relevant sources.

Each site also produced a final single STT system by applying standard ROVER or consensus network combination (CNC) methods to the individual systems described above.

3.7.4.3.3. Machine Translation

Machine translation experiments were carried out using the BBN HierDec MT system (Shen *et al.* 2008), which employs hierarchical rules to translate the source string to a target dependency tree. HierDec produces an N -best of translation hypotheses, which is subsequently rescored with a 5-gram target language model (LM) in order to choose the best answer. In the experiments carried out, the Arabic-English MT system was trained on approximately 200 million words of parallel training data; the Chinese-English MT was trained on about 230 million words of parallel text. In both cases, the English 5-gram LM was trained on approximately 8 billion words of text.

In all experiments, the MT system parameters were optimized on translations of the audio tuning set reference transcripts, rather than on STT output. This means that the same MT decoding weights were used when translating different STT outputs on the validation test sets.²⁰ The MT optimization criterion was the difference between the TER and BLEU scores.

3.7.4.4 Results

Table 3.106 and Table 3.107 compare various system combination configurations on the Arabic and Chinese AGILE development sets, respectively. STT was evaluated based on character error rate (CER), word error rate (WER), and word deletion rate (WDR). MT performance was measured in terms of TER and BLEU scores. In all cases,

²⁰ It could be argued that different STT outputs would require different MT tuning, however we did not find that to be true in our experiments.

STT was performed on manually segmented audio, and the STT output was post-processed to convert spoken numbers to written form prior to machine translation.

On Arabic, the lowest WER site-specific STT output was from BBN, produced by applying standard ROVER to the four individual BBN STT outputs described in the previous section. In Table 3.106 this output is compared to three AGILE-wide ROVER combinations. The first one is a three way standard ROVER, combining the (already combined) site-specific STT outputs from BBN, CU, and LIMSI. It can be seen that this three way ROVER gives a significant gain in WER and small gains in TER and BLEU compared to the standalone BBN ROVER. The next two AGILE-wide ROVER combinations shown in the table are applied to the 12 individual STT systems from BBN, CU, and LIMSI. The first such combination uses standard ROVER, while the second optimizes the ROVER parameters (alpha, null confidence score, and system weights) on the AGILE tuning set in order to minimize the function $WER + 0.6 * WDR$. The motivation for including WDR in the optimization criterion was based on earlier experiments in which a correlation between WDR and BLEU was observed. Basically, lower WDR tends to lead to higher BLEU scores after MT. This makes sense because the BLEU score includes a penalty factor that is a function of the MT hypothesis and reference lengths. If the STT WDR is high, then the STT output is likely to be shorter than normal, leading to deletions in the MT hypothesis and hence a more severe brevity penalty term in BLEU scoring. The results show that the standard 12-way ROVER outperforms the 3-way ROVER in terms of WER and WDR, and is also slightly better in TER and BLEU. The optimized 12-way ROVER can achieve even lower WER without increasing deletion rate, but the effect on MT is too small and inconsistent.

On Chinese, the lowest CER STT output was the CNC of the four individual systems from CU. In Table 3.107 this output is compared to four AGILE-wide system combinations. The first one is generated by cross-adapting each of CU's 4 STT systems to the BBN ROVER and LIMSI outputs separately, yielding a total of eight new STT outputs, and then combining those using CNC. It can be seen that cross-adaptation gives a significant gain in both CER and WER, and small improvements in TER compared to the standalone CU CNC STT output. The next two AGILE STT outputs are produced by applying standard word-based or character-based ROVER to the 11 individual STT systems from BBN, CU, and LIMSI.²¹ There is very little difference between these two and the cross-adapted output in terms of MT performance, even though the standard ROVER combinations tend to reduce CER and WDR. Finally, WDR can be reduced a bit further by optimizing the character-based ROVER (alpha, null confidence score, system weights) so as to minimize $WER + 0.6 * WDR$. However, this has no effect on MT.

²¹ Note that the character-based ROVER outputs a character sequence which is then segmented into words using the same tokenization process as the one applied to the MT parallel training data.

Test Set	System	STT		MT	
		WER	WDR	TER	BLEU
at1	BBN	12.4	2.2	54.5	26.7
	AGILE 3-way ROVER	11.3	2.2	54.3	27.1
	AGILE 12-way ROVER	11.3	1.8	54.3	27.3
	AGILE 12-way ROVER Opt.	10.9	1.8	54.2	27.3
at2	BBN	10.2	1.6	51.0	28.6
	AGILE 3-way ROVER	9.3	1.5	50.7	28.9
	AGILE 12-way ROVER	9.1	1.2	50.5	29.1
	AGILE 12-way ROVER Opt.	8.9	1.2	50.6	29.0

Table 3.106: Arabic STT/MT results on the AGILE validation test sets, comparing site-specific and cross-site system combination in terms of word error rate (WER), word deletion rate (WDR), translation edit rate (TER), and bilingual evaluation understudy (BLEU) scores.

Test Set	System	STT			MT	
		CER	WER	WDR	TER	BLEU
ct1	CU	9.4	14.2	2.7	66.4	19.2
	AGILE cross-adapted	8.7	13.0	2.9	66.0	19.2
	AGILE word ROVER	8.6	13.0	2.8	66.1	19.2
	AGILE char ROVER	8.5	13.0	2.6	66.1	19.2
	AGILE char ROVER Opt.	8.5	13.0	2.5	66.2	19.3
ct2	CU	8.2	11.7	2.3	66.0	17.1
	AGILE cross-adapted	7.4	10.5	2.5	65.8	17.2
	AGILE word ROVER	7.3	10.4	2.4	65.6	17.3
	AGILE char ROVER	7.2	10.5	2.3	65.7	17.3
	AGILE char ROVER Opt.	7.2	10.4	2.2	65.7	17.3

Table 3.107: Chinese STT/MT results on the AGILE validation test sets, comparing site-specific and cross-site system combination in terms of character error rate (CER), word error rate (WER), word deletion rate (WDR), translation edit rate (TER), and bilingual evaluation understudy (BLEU) scores.

Based on the above results, it is clear that a reduction in CER or WER does not guarantee improvement in MT accuracy. It would be preferable to optimize the STT ROVER combination using the method outlined in Section 3.7.4.2.3. A prototype of this algorithm was implemented using the BBN phrase-based MT system to translate the STT ROVER consensus network. Preliminary results on the AGILE Arabic tuning set show that this method can achieve a small gain (0.3% BLEU) in phrase-based MT after several (10 - 12) iterations of optimization. However, when this MT-optimized STT ROVER output is compared to standard or WER-optimized ROVER in terms of the more advanced BBN HierDec MT performance, there is no clear benefit. This result is slightly discouraging, but we are hoping that we can see a benefit in the future, when the tuning of the STT ROVER combination is performed based on the more accurate HierDec MT engine.

References

- Aertsen, A.M.H.J., P.I.M. Johannesma and D.J. Hermes. 1980. Spectro-temporal receptive fields of auditory neurons in the grassfrog. 2: Analysis of the stimulus-event relation for tonal stimuli. *Biological Cybernetics*, 38(4): pp. 235–248, November.
- Afify, M., L. Nguyen, B. Xiang, S. Abdou and J. Makhoul. 2005. Recent progress in Arabic broadcast news transcription at BBN. *Proceedings of InterSpeech*, Lisbon, Portugal, September.
- Akita, Y., and T. Kawahara. 2004. Language model adaptation based on PLSA of topics and speakers. *Proceedings of ICSLP*, pp. 1045-1048.
- Alexandrescu, A. and K. Kirchhoff. 2006. Factored neural language models. *NAACL*, June.
- Allen, J.B. 2005. *Articulation and Intelligibility*. Morgan and Claypool, Publisher.
- Allen, J. B. 1994. How do humans process and recognize speech? *IEEE Transactions on Speech and Audio* 2(4): pp. 567-577.
- Al-Onaizan, Y. and L. Mangu. 2007. *Arabic ASR and MT Integration for GALE*, *Proceedings of ICASSP*, Honolulu, Hawaii, USA.
- Al-Onaizan, Y. and Papineni, K. 2006. Distortion Models for Statistical Machine Translation. *21st International Conference on Computational Linguistics*.
- Altun, Y., I. Tsochantaridis and T. Hofmann. 2003. Hidden Markov support vector machines. *International Conference on Machine Learning (ICML)*, Washington, DC, USA.
- Amengual, J.C., J. M. Benedi, F. Casacuberta, A. Castaño, A. Castellanos, D. Llorens, A. Marzal, F. Pratand, E. Vidaland and J. M. Vilar. 1997. Using categories in the EuTrans system, *Proceedings of Spoken Lang. Translation Workshop*, Madrid, Spain, pp. 44–53.
- Asanovic, K., R. Bodik, J. Demmel, T. Keaveny, K. Keutzer, J. D. Kubiatowicz, E. A. Lee, N. Morgan, G. Necula, D. A. Patterson, K. Sen, J. Wawrzynek, D. Wessel and K. A. Yelick. 2008. The Parallel Computing Laboratory at U.C. Berkeley: A Research Agenda Based on the Berkeley View, EECS Department, University of California, *Berkeley Technical Report* No. UCB/EECS-2008-23, March 21.
- Athineos, M., H. Hermansky and D. P.W. Ellis. 2004. LP-TRAP: Linear predictive temporal patterns. *ICSLP'04*, pp. 1154–1157, Jeju, KR.
- Bacchiani, M., M. Riley, B. Roark and R. Sproat. 2006. MAP Adaptation of Stochastic Grammars. *Computer Speech & Language*, 20(1): pp. 41-68.
- Bahl, L., P. F. Brown, P. V. de Souza and R. L. Mercer. 1986. Maximum Mutual Information Estimation of Hidden Markov Model Parameters for Speech Recognition. *Proceedings of ICASSP'86*.

- Bellegarda, J. 2000. Exploiting latent semantic information in statistical language modeling. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 88(8): pp. 63-75, Aug.
- Bellegarda, J. 2005. Latent semantic mapping: Dimensionality reduction via globally optimal continuous parameter modeling. *Proceedings of IEEE ASRU Workshop*, pp. 127- 132.
- Bender, O., E. Matusov, S. Hahn, S. Hasan, S. Khadivi and H. Ney. 2007. The RWTH Arabic-to-English spoken language translation system, *Proceedings of ASRU' 2007*, Kyoto.
- Bengio, Y., R. Ducharme and P. Vincent. 2001. A neural probabilistic language model. *Advances in Neural Information Processing Systems*.
- Bertoldi, N., R. Zens, M. Federico and W. Shen. 2008. Efficient Speech Translation Through Confusion Network Decoding, *IEEE Transactions on Audio, Speech and Language*, Vol. 16, No. 8, pp. 1696-1705.
- Bertholdi, N. and M. Federico. 2005. A new decoder for spoken language translation based on confusion networks, *ASRU-05*.
- Billa, J., M. Noamany, A. Srivastava, D. Liu, R. Stone, J. Xu, J. Makhoul and F. Kubala. 2002. Audio Indexing of Arabic Broadcast News. *Proceedings of ICASSP*, pp. 5-8. April.
- Bilmes, J., K. Asanovi, C. Chin and J. Demmel. 1997. Using phipac to speed error back-propagation learning. *Proceedings of ICASSP*.
- Bisani, M., and H. Ney. 2008. Joint-Sequence Models for Grapheme-to-Phoneme Conversion. *Speech Communication*, 50: pp. 434 – 451.
- Bisani, M. and H. Ney. 2005. Open vocabulary speech recognition with flat hybrid models. *Proceedings of European Conf. on Speech Communication and Technology*, pp. 725 – 728, Lisbon, Portugal, September.
- Bisani, M., and Ney, H. 2003. Multigram-based grapheme-to-phoneme conversion for LVCSR, *Proceedings of EUROSPEECH*, vol. 2, pp. 933-936.
- Blei, D., A. Ng, and M. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research*, pp. 1107-1135.
- Boulevard, H. and S. Dupont. 1996. A new ASR approach based on independent processing and re-combination of partial frequency bands. *Proceedings of ICSLP*.
- Boulevard, H. and Morgan N. 1994. *Connectionist Speech Recognition - A Hybrid Approach*. Kluwer Academic Publishers.
- Brants, T., C. Ashok, P. Popat, F. Xu, J. Och and J. Dean. 2007. Large language models in machine translation. *EMNLP*, pp. 858–867.

- Brown, P.F., S. A. Della Pietra, V. J. Della Pietra and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation, *Comput. Linguist.*, vol. 19, no. 2, pp. 263–312.
- Buckwalter, T. 2004. LDC2004L02: Buckwalter Arabic morphological analyzer version 2.0.
- Buckwalter, T. 2002. Buckwalter Arabic Morphological Analyzer (LDC2002L49, LDC2004L02). <http://www.qamus.org/morphology.htm>. Linguistic Data Consortium, Philadelphia.
- Chan, H.Y., and P.C. Woodland. 2004. Improving broadcast news transcription by lightly supervised discriminative training. *ICASSP*, Montreal.
- Chang, P.C., S.P. Liao, and L-S. Lee. 2003. Improved Chinese broadcast news transcription by language modeling with temporally consistent training corpora and iterative phrase extraction. *Proceedings of Eurospeech*, pp. 421–424.
- Chen, L., J-L. Gauvain, L. Lamel and G. Adda. 2004. Dynamic Language Modeling for Broadcast News. *Proceedings of ICSLP'04*.
- Chen, S., B. Kingsbury, L. Mangu, D. Povey, G. Saon, H. Soltau and G. Zweig. 2006. Advances in Speech Transcriptions at IBM under the DARPA EARS Program. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5): pp. 1569-1608.
- Chen, S.F. 2003. Conditional and Joint Models for Grapheme-to-Phoneme Conversion. *Proceedings European Conf. on Speech Communication and Technology*, pages 2033 – 2036, Geneva, Switzerland, September.
- Chi, T., Y. Gao, M.C. Guyton, P. Ru and S.A. Shamma. 1999. Spectro-temporal modulation transfer functions and speech intelligibility, *J. Acoust. Soc. Am.*, 106(5): pp. 719- 2732.
- Chien, J.T. and C. H. Chueh. 2008. Latent Dirichlet language model for speech recognition. *IEEE Workshop on Spoken Language Technology*, pp. 201-204.
- Choueiter, G., D. Povey, F. C. Stanley and G. Zweig. 2006. Morpheme-Based Language Modeling for Arabic LVCSR. *Proceedings of ICASSP*, Toulouse, France.
- Chu, S.M., H-K. Kuo, Y. Y. Liu, Y. Qin, Q. Shi and G. Zweig. 2007. The IBM Mandarin Broadcast Speech Transcription System. *Proceedings of ICASSP'07*.
- Cohn, D., L. Atlas and R. Ladner. 1994. Improving generalization with active learning. *Machine Learning*, 15: pp. 201–221.
- Darroch, J.N., and D. Ratcliff. 1972. Generalized iterative scaling for log-linear models. *The Annals of Mathematical Statistics*, 43(5): pp. 1470-1480.
- Dau, T., B. Kollmeier and A. Kohlrausch. 1997. Modeling auditory processing of amplitude modulation .I Detection and masking with narrow-band carriers. *Journal Acoustic Society of America* 102.

- de Cheveigné, A. and H. Kawahara. 2002. YIN, A Fundamental Frequency Estimator for Speech and Music. *Journal of the Acoustical Society of America*, 111: pp. 1917-1930.
- Deerwester, S.C., S. T. Dumais, T. K. Landauer, G. W. Furnas and R. A. Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6): pp. 391--407.
- Deligne, S. and F. Bimbot. 1995. Language modeling by variable length sequences: Theoretical formulation and evaluation of multigrams. *Proceedings of ICASSP*, pp. 169 – 172, Detroit, Michigan, USA, May.
- Deligne, S., F. Yvon and F. Bimbot. 1995. Variable length sequence matching for phonetic transcription using joint multigrams. *Proceedings European Conf. on Speech Communication and Technology*, pp. 2243 –2246, Madrid, Spain, September.
- Deng, Y. and W.J. Byrne. 2008. HMM word and phrase alignment for statistical machine translation, *IEEE Trans. ASLP*, Vol. 16, pp. 494-507.
- Despres, J., P. Fousek, J-L. Gauvain, S. Gay, Y. Josse, L. Lamel and A. Messaoudi. 2008. The LIMSI-Vecsys Research Systems for NBest 2008. *N-Best workshop*, Soesterberg, NL.
- Doddington, G. 2002. Automatic evaluation of MT quality using n-gram co-occurrence statistics. *Proceedings of human language technology conference*, pp. 138-145.
- Duda, R.O., P. E. Hart and D. G. Stork. 2001. *Pattern Classification*. John Wiley and Sons, Inc., second edition.
- Dyer, C. and S. Muresan and P. Resnik 2008. Generalizing word lattice translation. *46th Annual Meeting of the ACL*.
- Ellis, D.P.W., R. Singh and S. Sivasdas. 2001. Tandem Acoustic Modeling in Large-Vocabulary Recognition. *Proceedings of ICASSP'01*.
- Emami, A. and F. Jelinek. 2005. A neural syntactic language model. *Machine Learning*, 60(1-3):195–227.
- Emami, A., P. Xu and F. Jelinek. 2003. Using a connectionist model in a syntactical based language model. *Proceedings of ICASSP*.
- Evermann, G. and P. C. Woodland. 2000. Posterior probability decoding, confidence estimation and system combination. *Proceedings of the NIST Speech Transcription Workshop*.
- Faria, A. and N. Morgan. 2008. Corrected Tandem Features for Acoustic Model Training, *Proceedings of ICASSP*.
- Finke, M. and A. Waibel. 1997. Speaking mode dependent pronunciation modeling in large vocabulary conversational speech recognition. *Proceedings of Eurospeech*, volume 5, pp. 2379–2382.
- Finke, M., P. Geutner, H. Hild, T. Kemp, K. Ries and M. Westphal. 1997. The Karlsruhe Verbomobil Speech Recognition Engine, *Proceedings of ICASSP*, Germany.

- Fiscus, J. G. 1997. A post-processing system to yield reduced word error rates: Recogniser output voting error reduction (ROVER). *IEEE Workshop on ASRU*, pp. 347–352, Santa Barbara, CA.
- Fletcher, H. 1995. The ASA edition of Speech and Hearing in Communication. *Journal Acoustic Society of America*.
- Fousek, P., L. Lamel and J-L. Gauvain. 2008. On the use of MLP features for broadcast news transcription. *TSD'08*, volume 5246 of *Lecture Notes in Computer Science*, pages 303–310.
- Fousek, P., L. Lamel and J-L. Gauvain. 2008a. Transcribing Broadcast Data Using MLP Features. *Interspeech'08*, pp. 1433–1436, Brisbane, AU.
- Fousek, P. 2007. Extraction of features for automatic recognition of speech based on spectral dynamics. *Ph.D. thesis, Czech Technical University in Prague, Faculty of Electrical Engineering, Prague*.
- Fügen, C. and I. Rogina. 2000. Integrating dynamic speech modalities into context decision trees. *Proceedings of ICASSP*.
- Gadde, V.R.R. 2000. Modeling word durations, *Proceedings of ICSLP*, vol. 1, pp. 601–604, October.
- Gales, M.J.F., X. Liu, R. Sinha, P. C. Woodland, K. Yu, S. Matsoukas, T. Ng, K. Nguyen, L. Nguyen, J. L. Gauvain, L. Lamel and A. Messaoudi. 2007. Speech recognition system combination for machine translation. *Proceedings of ICASSP*, volume 4, pp. 1277–1280.
- Gales, M.J.F., B. Jia, X. Liu, K. C. Sim, P. C. Woodland and K. Yu. 2005. Development of the CUHTK 2004 Mandarin conversational telephone speech transcription system. *Proceedings of ICASSP*, Philadelphia.
- Gales, M.J.F. 1999. Semi-tied covariance matrices for hidden Markov models. *IEEE Trans.Speech and Audio Processing*, 7(3), May.
- Gales, M.J.F. 1998. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech and Language*, 12: pp. 75–98.
- Gales, M.J.F. 1997. Maximum likelihood linear transformation for hmm-based speech recognition. *Tech. Report CUED/FINFENG/TR291*, Cambridge University Engineering Dept.
- Gauvain, J-L, L. Lamel and Gilles Adda. 2002. The LIMSI Broadcast News Transcription System. *Speech Communication*, 37(1-2): pp. 89–108.
- Gauvain, J-L., L. F. Lamel, G. Adda and M. Adda-Decker. 1994. Speaker-independent continuous speech dictation. *Speech Communication*, 15: pp. 21–37.
- Gelbart, D. 2008. Ensemble Feature Selection for Multi-stream Automatic Speech Recognition. *Ph.D. Thesis, University of California at Berkeley*.

- Gildea, D. and T. Hofmann. 1999. Topic-based language models using EM. *Proceedings of European Conference on Speech Communication and Technology*.
- Gillick, L. and S. Cox. 1989. Some statistical issues in the comparison of speech recognition algorithms. *Proceedings of the ICASSP*, pp. 532-535.
- Grézl, F. and P. Fousek. 2008. Optimizing bottle-neck features for LVCSR. *ICASSP'08*, pp. 4729–4732.
- Greenwood, D.D. 1990. A cochlear frequency-position function for several species—29 years later. *Acoustical Society of America Journal*, 87: pp. 2592–2605, June.
- Grézl, F., M. Karafiát, S. Kontár and J. Černocký. 2007. Probabilistic and bottle-neck features for LVCSR of meetings. *Proceedings of ICASSP*, pp. 757–760.
- Guiliani, D. and F. Brugnara. 2006. Acoustic model adaptation with multiple supervisions. *Proceedings TCSTAR Workshop on Speech-to-Speech Translation*, pp. 151–154, Barcelona, Spain, June.
- Habash, N. and R. Owen. 2005. Arabic tokenization, morphological analysis, and part-of-speech tagging in one fell swoop. *Proceedings of the Association for Computational Linguistics*, pp. 573 – 580, Ann Arbor, Michigan, USA, June.
- Hain, T. *et al.* 2005. The 2005 AMI System for the Transcription of Speech in Meetings. *NIST RT05 Workshop*, Edinburgh, UK.
- Hatch, A.O., S. Kajarekar and A. Stolcke. 2006. Within-class covariance normalization for svmbased speaker recognition. *Proceedings of ICSLP*, pp. 1471–1474.
- Heidel, A., H. Chang and L. Lee. 2007. Language model adaptation using latent Dirichlet allocation and an efficient topic inference algorithm. *Proceedings of Interspeech*, pp. 2361– 2364.
- Heigold, G., R. Schlüter and H. Ney. 2009. Modified MMI/MPE in a transducer-based framework. *Proceedings of ICASSP*, Taipei, Taiwan.
- Heigold, G., T. Deselaers, R. Schlüter and H. Ney. 2008a. Modified MMI/MPE: A direct evaluation of the margin in speech recognition. *International Conference on Machine Learning (ICML)*, Helsinki, Finland.
- Heigold, G., P. Lehnen, R. Schlüter and H. Ney. 2008b. On the equivalence of Gaussian and log-linear HMMs. *ICLSP (Interspeech)*, Brisbane, Australia.
- Hermansky, H. and P. Fousek. 2005. Multiresolution RASTA filtering for TANDEM-based ASR. *Interspeech'05*, pp. 361–364, Lisboa, PT.
- Hermansky, H., D.P.W. Ellis and S. Shamma. 2000a. Tandem connectionist feature extraction for conventional HMM systems, *Proceedings of ICASSP, 2000*.
- Hermansky, H., D. Ellis and S. Shamma. 2000b. Connectionist feature extraction for conventional HMM systems. *Proceedings of ICASSP 2000*.

- Hermansky, H. and S. Shamma. 1998. TRAPS Classifiers of temporal patterns, *Proceedings of ICSLP-98*, Sydney, vol. 3, pp. 1003-1006.
- Hermansky, H. 1998. Should recognizers have ears? *Speech Communication* 25: pp. 3-27.
- Hermansky, H. and S. Tibrewala. 1997. Sub-band based recognition of noisy speech. *Proceedings of ICASSP 1997*.
- Hermansky, H., S. Tibrewala and M. Pavel. 1996. Towards ASR on partially corrupted speech. *Proceedings of ICSLP 1996*.
- Hermansky, H. and N. Morgan. 1994. RASTA processing of speech. *IEEE Trans. on Speech and Audio Proc* 2: pp. 578-589.
- Hermansky, H. 1990. Perceptual linear predictive (plp) analysis of speech. *Journal of the Acoustical Society of America*, 87(4): pp. 1738–1752.
- Hildebrand, A., K. Rottmann, T. Notari, Q. Gao, S. Hewavitharana, N. Bach and S. Vogel. 2008. Recent improvements in the CMU large-scale chinese-english SMT system. *The joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics*.
- Hoffmeister, B., C. Plahl, P. Fritz, G. Heigold, J. Löff, R. Schlüter, and H. Ney. 2007. Development of the 2007 RWTH Mandarin GALE LVCSR system, *Proceedings of ASRU*, pp. 455–460.
- Hoffmeister, B., T. Klein, R. Schlüter and H. Ney. 2006. Frame based system combination and a comparison with weighted ROVER and CNC. *Proceedings of ICSLP*.
- Honal, M. and T. Schultz. 2005. Automatic Disfluency Removal on Recognized Spontaneous Speech – Rapid Adaptation to Speaker-Independent Disfluencies, *Proceedings of ICASSP*, Philadelphia, Pennsylvania, USA.
- Houtgast, T. 1989. Frequency selectivity in amplitude modulation detection. *Journal Acoustic Society of America*, 88.
- Hsu, B.J. and J. Glass. 2006. Style and topic language model adaptation using HMM-LDA. *Proceedings of Empirical Methods on Natural Language Processing*.
- Hwang, M.Y., G. Peng, W. Wang, A. Faria and A. Heidel. 2007. Building a highly accurate Mandarin speech recognizer with language-independent technologies and language-dependent modules. *Proceedings of ASRU*.
- Hwang, M.Y., G. Peng, M. Ostendorf, W. Wang, A. Faria and A. Heidel. 2009. Building a highly accurate Mandarin speech recognizer with language-independent technologies and language-dependent modules. *IEEE Trans. on Audio, Speech, and Language Processing*. To appear.
- Iyer, R. and M. Ostendorf. 1999. Modeling long distance dependence in language: Topic mixtures versus dynamic cache models. *IEEE Transactions on Speech and Audio Processing*, 7(1): pp. 30--39, Jan.

- Janin, A., A. Stolcke, X. Anguera, K. Boakye, O. Cetin, J. Frankel and J. Zheng. 2006. The ICSI-SRI Spring 2006 Meeting Recognition System, S. Renals and S. Bengio, editors, *Machine Learning for Multimodal Interaction: Third International Workshop (MLMI 2006)*, Lecture Notes in Computer Science. Springer.
- Jebara, T. 2002. Discriminative, generative, and imitative learning. *Doctoral dissertation, Massachusetts Institute of Technology*.
- Jin, Q. and T. Schultz. 2004. Speaker segmentation and clustering in meetings. *Proceedings of the ICSLP*.
- Juang, B.-H. and S. Katagiri. 1992. Discriminative learning for minimum error classification, *IEEE Trans. Signal Process.* 40 (12) (1992), pp. 3043–3054.
- Kamm, T.M. and G.G.L. Meyer. 2002. Selective sampling of training data for speech recognition. *Proceedings Human Language Technology*, San Diego.
- Kanedera, N., T. Arai, H. Hermansky and M. Pavel. 1999. On the relative importance of various components of the modulation spectrum for automatic speech recognition, *Speech Communication*, 28: pp. 43-55.
- Kanthak, S., D. Vilar, E. Matusov, R. Zens and H. Ney. 2005. Novel reordering approaches in phrase-based statistical machine translation. *ACL Workshop on Building and Using Parallel Texts: Data-Driven Machine Translation and Beyond*.
- Kanthak, S. and H. Ney. 2001. Context-Dependent Acoustic Modeling Using Graphemes for Large Vocabulary Speech Recognition. *Proceedings of ICASSP*, pp. 845 – 848.
- Karypis, G. 2002. Cluto. <http://glaros.dtc.umn.edu/gkhome/views/cluto/>.
- Kenny, P., P. Ouellet, N. Dehak and V. Gupta. 2008. A study of Interspeaker Variability in Speaker Verification. *IEEE Trans. on Audio, Speech and Language Processing*, 16(5): pp. 980–987.
- Kemp, T. and A. Waibel. 1997. Unsupervised Training of a Speech Recognizer Using TV Broadcasts. *Proceedings of ICSLP*.
- Kim, W. 2004. Language model adaptation for automatic speech recognition and statistical machine translation. *Ph.D. thesis, Johns Hopkins University*. Distributed by JHU.
- Kingsbury, B., N. Morgan and S. Greenberg. 1998. Robust speech recognition using the modulation spectrogram, *Speech Communication*, vol. 25, no. 1-3, pp. 117-132, Aug.
- Kirchhoff, K., D. Vergyri, J. Bilmes, K. Duh and A. Stolcke. 2006. Morphology-based language modeling for conversational Arabic speech recognition. *Computer Speech and Language*, 20(4).
- Kirchhoff, K., J. Bilmes, S. Das, N. Duta, M. Egan, J. Gang, H. Feng J. Henderson, L. Daben, M. Noamany, P. Schone, R. Schwartz and D. Vergyri. 2003. Novel approaches to Arabic speech recognition: report from the 2002 JHU Summer Workshop, *Proceedings of ICASSP*, vol.1, pp. 344–347, April.

- Kleinschmidt, M. 2003. Localized spectro-temporal features for automatic speech recognition, *Proceedings of Eurospeech*, pp. 2573-2576.
- Kneser, R. and H. Ney. 1995. Improved backing-off for M-gram language modeling. *Proceedings of ICASSP*, volume 1, pp. 181-184.
- Kneser, R., J. Peters and D. Klakow. 1997. Language model adaptation using dynamic marginals. *Proceedings of European Conference on Speech Communication and Technology*, pp. 1971-1974.
- Koehn, P., F. Och and D. Marcu. 2003. Statistical phrase-based translation, *NAACL/HLT*.
- Kozhevnikov, V.A. and L.A. Chistovich. 1967. *Speech: Articulation and Perception*, translated from Russian by U.S. Dept. of Commerce, pp. 250-251.
- Kumar, S., Y. Deng and W.J. Byrne. 2006. A weighted finite state transducer translation template model for statistical machine translation, *Natural Language Engineering*, Vol. 12, pp. 35-75.
- Kumar, S., & W.J. Byrne. 2005. Local phrase reordering models for statistical machine translation, *Proceedings of HLT and EMNLP*.
- Lamel, L., A. Messaoudi and J-L Gauvain. 2008. Investigating Morphological Decomposition for Transcription of Arabic Broadcast News and Broadcast Conversation Data. *Interspeech'08*, pp. 1429-1432, Brisbane, Australia.
- Lamel, L., A. Messaoudi and J-L Gauvain. 2007. Improved Acoustic Modeling for Transcribing Arabic Broadcast Data. *Interspeech'07*, pp. 2077-2080, Antwerp, BE.
- Lamel, L., J-L. Gauvain and G. Adda. 2002. Lightly supervised and unsupervised acoustic model training. *Computer Speech and Language*, 16: pp. 115-129.
- Leggetter, C. and P. Woodland. 1995. Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. *Computer Speech & Language*, 9(2): pp. 171-185, April.
- Lei, X., et. al. 2006. Improved tone modeling for Mandarin broadcast news speech recognition. *Proceedings ICSLP*, pp. 1237-1240, September.
- Li, F. and J.B. Allen. 2009. Multiband product rule and consonant identification *Journal Acoustic Society of America*.
- Li, J., Z. Yan, C. Lee and R. Wang. 2007. A study on soft margin estimation for LVCSR. *IEEE ASRU*, Kyoto, Japan.
- Liu, Y. and F. Liu. 2008. Unsupervised language model adaptation via topic modeling based on named entity hypotheses. *Proceedings of ICASSP*.
- Liu, Y., E. Shriberg, A. Stolcke, B. Peskin, J. Ang, D. Hillard, M. Ostendorf, M. Tomalin, P. C. Woodland and M. Harper. 2005. Structural Metadata Research in the EARS Program, *Proceedings of the ICASSP*.

Liu, X.A., M. J. F. Gales and P. C. Woodland. 2008. Context dependent language model adaptation. *Proceedings of Interspeech'08*.

Ma, J., S.Matsoukas, O. Kimball and R. Schwartz. 2006. Unsupervised training on large amount of broadcast news data. *Proceedings ICASSP*, Toulouse.

Mangu, L., E. Brill and A. Stolcke. 1999. Finding consensus among words: lattice-based word error minimization. *Eurospeech*.

Mangu, L., E. Brill and A. Stolcke. 2000. Finding Consensus in Speech Recognition: Word Error Minimization and Other Applications of Confusion Networks. *Computer, Speech and Language*, 14(4): pp. 373-400.

Mathias, L. and W. Byrne. 2006. Statistical phrase-based speech translation. *ICASSP-06*.

Matusov, E., B. Hoffmeister and H. Ney 2008. ASR Word Lattice Translation with Exhaustive Reordering is possible, *Interspeech-08*.

Matusov, E., S. Kanthak and H. Ney 2005. On the integration of speech recognition and statistical machine translation, *Interspeech-05*.

McDermott, E. 1997. Discriminative Training for Speech Recognition, *Ph.D. Thesis, Waseda*.

Messaoudi, A., J-L Gauvain and L. Lamel. 2005. Modeling Vowels for Arabic BN Transcription. *InterSpeech Eurospeech'05*, pp. 1633-1636, Lisbon. September.

Messaoudi, A., J-L Gauvain and L. Lamel. 2006. Arabic Broadcast News Transcription using a One Million Word Vocalized Vocabulary. *Proceedings of ICASSP*, pp. 1093-1096.

Miller, L. M., M. A. Escabi, H. L. Read and C. E. Schreiner. 2002. Spectrotemporal Receptive fields in the lemniscal auditory thalamus and cortex. *The journal of Neurophysiology* 87(1).

Misra, H., H. Bourlard and V. Tyagi. 2003. New Entropy Based Combination Rules In Hmm/Ann Multi-Stream ASR. *Proceedings of ICASSP'03*.

Mohri, M., F. Pereira & M. Riley. 2002. Weighted finite-state transducers in speech recognition, *Computer Speech and Language*. Vol. 16, pp. 69–88.

Morgan, N. and H. Bourlard. 1995. Continuous speech recognition: An introduction to the hybrid HMM/connectionist approach, *IEEE Signal Processing Magazine*, vol. 12, no. 3, pp. 25-42, May.

Morgan, N., Q. Zhu, A. Stolcke, K. Sonmez, S. Sivasdas, T. Shinozaki, M. Ostendorf, P. Jain, H. Hermansky, D. Ellis, G. Doddington, B. Chen, O. Cretin, H. Bourlard and M. Athineos. 2005. Pushing the envelope - aside [speech recognition], *IEEE Signal Processing Magazine*, vol. 22, no. 5, pp. 81-88, Sep.

Morgan, N., B. Y. Chen, Q. Zhu and A. Stolcke. 2004. TRAPping Conversational Speech: Extending TRAP/Tandem Approaches to Conversational Telephone Speech Recognition. *Proceedings of ICASSP'04*.

- Mrva, D. and P. C. Woodland. 2004. A PLSA-based language model for conversational telephone speech. *Proceedings of ICSLP*.
- Navratil, J. 2006. Recent advances in phonotactic language recognition using binary-decision trees. *Proceedings of Interspeech*.
- Ney, H. 1999. Speech translation: coupling of recognition and translation. *ICASSP-99*
- Ng, T., B. Zhang, K. Nguyen and L. Nguyen. 2008. Progress in the BBN 2007 Mandarin Speech to Text System. *Proceedings of ICASSP*.
- Ng, T., K. Nguyen, R. Zbib and L. Nguyen. 2009. Improved Morphological Decomposition for Arabic Broadcast News Transcription. *Proceedings of ICASSP*.
- Nguyen, L. and B. Xiang. 2004. Light supervision in acoustic model training. *Proceedings of ICASSP*, Montreal, Canada, May.
- Nguyen, L., S. Matsoukas, J. Davenport, F. Kubala, R. Schwartz and J. Makhoul. 2002. Progress in transcription of broadcast news using byblos. *Speech Communication*, 38: pp. 213–230.
- Nguyen, L. and R. Schwartz. 1997. Efficient 2-pass n-best decoder. *Proceedings of EuroSpeech*, pp. 167–170, Rhodes, Greece, September.
- Noamany, M., T. Schaaf and T. Schultz. 2007. Advances in the CMU/INTERACT Arabic GALE Transcription System, *Proceedings of HLT/NAACL*, USA.
- Och, F.J. 2003. Minimum error rate training in statistical machine translation. *The joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics*.
- Och, F. and H. Ney. 2000. Improved statistical alignment models, *38th Annual Meeting of the ACL*.
- Papineni, K., S. Roukos, T. Ward and W. Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. *The joint conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics*.
- Park, J., F. Diehl, M. J. F. Gales, M. Tomalin and P. C. Woodland. 2009. Training and adapting MLP features for Arabic speech recognition. *Proceedings of ICASSP*.
- Paulik, M., S. Rao, I. Lane, S. Vogel and T. Schultz. 2008. Sentence Segmentation and Punctuation Recovery for Spoken Language Translation, *Proceedings of ICASSP*.
- Pavel, M., M. Slaney and H. Hermansky. 2009. Reconciliation of human and machine speech recognition performance. *Proceedings of ICASSP 2009*.
- Plahl, C., B. Hoffmeister, M.Y. Hwang, D. Lu, G. Heigold, J. Löff, R. Schlüter and H. Ney. 2008. Recent improvements of the RWTH GALE Mandarin LVCSR system. *Proceedings Interspeech*, pp. 2426–2429.
- Povey, D., D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon and K. Visweswariah. 2008. Boosted MMI for Feature and Model Space Discriminative Training. *ICASSP*.

- Povey, D., S. M. Chu and B. Varadarajan. 2008a. Universal Background Model Based Speech Recognition. *ICASSP*.
- Povey, D. and B. Kingsbury. 2007. Evaluation of Proposed Modifications to MPE for Large Scale Discriminative Training. *Proceedings of ICASSP'2007*.
- Povey, D., B. Kingsbury, L. Mangu, G. Saon, H. Soltau and G. Zweig. 2005. fMPE: Discriminatively Trained Features for Speech Recognition. *Proceedings of ICASSP*, Vol. 1 pp961-964.
- Povey, D. 2005. Improvements to fMPE for discriminative training of features. *Interspeech 2005*.
- Povey, D. 2004. Discriminative Training for Large Vocabulary Speech Recognition. PhD thesis, Cambridge University.
- Povey, D., M.J.F. Gales and P.C. Woodland. 2003. Discriminative MAP for acoustic model adaptation. *Proceedings of ICASSP*, Hong Kong, China.
- Povey, D. and P. Woodland, 2002. Minimum Phone Error and I-smoothing for Improved Discriminative Training. *Proceedings of ICASSP*, pp. 105–108.
- Povey, D., B. Kingsbury, L. Mangu, G. Saon, H. Soltau and G. Zweig. 2005. fMPE: Discriminatively trained features for speech recognition. *Proceedings of ICASSP*.
- Povey, D. 2009. Subspace Gaussian Mixture Models for Speech Recognition, Microsoft Research technical report. MSR-TR-2009-64
- Povey, D., L. Burget, et al. 2010. "Subspace Gaussian Mixture Models for Speech Recognition." *Proceedings of ICASSP 2010*.
- Powell, M.J.D. 1964. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *The Computer Journal*, pp. 155–162.
- Reichl, W. and G. Ruske. 1995. Discriminative training for continuous speech recognition. *Proceedings of EUROSPEECH-1995*, pp. 537-540
- Press, W.H., S. A. Teukolsky, W. T. Vetterling and B. P. Flannery. 1992. *Numerical Recipes*, chapter 10.5. Cambridge University Press.
- Rao, S., I. Lane and T. Schultz. 2007a. Optimizing Sentence Segmentation for Speech Translation, *Proceedings of Interspeech*, Antwerp, Belgium.
- Rao, S., I. Lane and T. Schultz. 2007b. Improving Spoken Language Translation by Automatic Disfluency Removal: Evidence from Conversational Speech Transcripts, *Proceedings of Machine Translation Summit*, Copenhagen, Denmark.
- Reichl, W. and W. Chou. 1999. A Unified Approach of Incorporating General Features in Decision Tree Based Acoustic Modeling. *Proceedings of ICASSP*.
- Reynolds, D. A, T. F. Quatieri and R. Dunn. 2000. Speaker verification using adapted Gaussian mixture models. *Digital Signal Processing*, 10(1-3): pp. 19–41.

- Riedmiller, M. and H. Braun. 1993. A direct adaptive method for faster backpropagation learning: The Rprop algorithm. *International Conference on Neural Networks (ICNN)*, San Francisco, CA, USA.
- Rottmann, K. and S. Vogel. 2007. Word reordering in statistical machine translation with a POS-based distortion model. *Proceedings of the 11th International Conference on Theoretical and Methodological Issues in Machine Translation*.
- Rybach, D., S. Hahn, C. Gollan, R. Schlüter and H. Ney. 2007. Advances in Arabic broadcast news transcription at RWTH, *Proceedings of ASRU*, pp. 449–454.
- Saon, G., D. Povey and H. Soltau. 2009. Large margin semi-tied covariance transforms for discriminative training. *Proceedings of ICASSP*, Taipei, Taiwan.
- Saon, G. and D. Povey. 2008. Penalty function maximization for large margin HMM training. *ICSLP (Interspeech)*, Brisbane, Australia.
- Saon, G. and M. Picheny. 2007. Lattice-based Viterbi decoding techniques for speech translation. *ASRU Workshop*.
- Saon, G., D. Povey and G. Zweig. 2005. Anatomy of an extremely fast LVCSR decoder. *Interspeech-05*.
- Saon, G., G. Zweig, B. Kingsbury, L. Mangu and U. Chaudhari. 2003. An architecture for rapid decoding of large vocabulary conversational speech. *Eurospeech*.
- Saon, G. and M. Padmanabhan. 2001. Data-driven approach to designing compound words for continuous speech recognition. *IEEE Trans. on Speech and Audio Processing*, 9(4):327–332, May.
- Saon, G., M. Padmanabhan, R. Gopinath and S. Chen. 2000. Maximum Likelihood Discriminant Feature Spaces. *Proceedings of ICASSP'00*, 2: pp. 1129-1132.
- Saon, G. and D. Povey and G. Zweig. 2005. Anatomy of an extremely fast LVCSR decoder. *Interspeech-05*.
- Sarikaya, R. and Y. Deng. 2007. Joint morphological-lexical language modeling for machine translation. *Human Language Technology conference*.
- Schlüter, R., I. Bezrukov, H. Wagner and H. Ney. 2007. Gammatone features and feature combination for large vocabulary speech recognition. *Proceedings of ICASSP*, volume 4, pp. 649–652, Honolulu, HI, USA, April.
- Schlüter, R., A. Zolnay and H. Ney. 2006. Feature combination using linear discriminant analysis and its pitfalls. *Proceedings of ICSLP*, pp. 345–348, Pittsburgh, PA, USA, September.
- Schwenk, H. and E. Yannick. 2008. Data selection and smoothing in an open-source system for the 2008 NIST machine translation evaluation. *Interspeech*, pp. 2727–2730.
- Schwenk, H. 2007. Continuous space language models. *Computer Speech and Language*, 21: pp. 492–518.

- Schwenk, H. and J-L. Gauvain. 2005a. Building continuous space language models for transcribing european languages. *Interspeech*, pp. 737–740.
- Schwenk, H. and J-L. Gauvain. 2005b. Training neural network language models on very large corpora. *EMNLP*, pp. 201–208.
- Schwenk, H. 2004. Efficient training of large neural networks for language modeling. *IJCNN*, pp. 3059–3062.
- Schwenk, H. and J-L. Gauvain. 2002. Connectionist Language Modeling for Large Vocabulary Continuous Speech Recognition. *Proceedings of ICASSP*, pp. 765–768.
- Sha, F. and L. Saul. 2007. Comparison of large margin training to other discriminative methods for phonetic recognition by hidden Markov models. *Proceedings of ICASSP*.
- Shamma, S. 1999. Multi-Stream Approach To Robust Speech Recognition. *PhD. Thesis, OGI*, Portland, USA.
- Shen, W., R. Zens, N. Bertoldi and M. Federico. 2006. On the integration of speech recognition and statistical machine translation. *IWSLT-06*.
- Shen, L., J. Xu and R. Weischedel. 2008. A new string to dependency machine translation algorithm with a target dependency language model. *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*.
- Shi, Q., S.M. Chu, W. Liu, H-K. Kuo, Y. Liu and Y. Qin. 2008. Search and classification based language model adaptation. *Proceedings of Interspeech*, pp. 1578–1581.
- Shriberg, E. and A. Stolcke. 2004. Prosody Modelling for Automatic Speech Recognition and Understanding, *Mathematical Foundations of Speech and Language Processing, IMA Volumes in Mathematics and Its Applications, Vol 138*, Springer-Verlag, New York: pp. 105-114.
- Shriberg, E.E. 1994. Preliminaries to a Theory of Speech Disfluencies. *PhD thesis, Berkeley's University of California*.
- Sinha, R., M. J. F. Gales, D. Y. Kim, X. A. Liu, K. C. Sim and P. C. Woodland. 2006. The CU-HTK Mandarin Broadcast News Transcription System. *Proceedings of ICASSP*.
- Solomonoff, A., W.M. Campbell and I. Boardman. 2005. Advances in channel compensation for svm speaker recognition. *ICASSP*, volume 1, pp. 629–632.
- Soltau, H. and A. Waibel. 2000. Phone dependent modeling of hyperarticulated effects. *Proceedings of ICSLP*.
- Soltau, H., F. Metze, C. Fügen and A. Waibel. 2001. A one-pass decoder based on polymorphic linguistic context assignment. *Proceedings of ASRU*.
- Soltau, H., G. Saon, D. Povey, L. Mangu, J. Kuo, M. Omar and G. Zweig. 2007. The IBM 2006 GALE Arabic ASR System. *Proceedings of ICASSP*, vol. 4, pp. 349–352.
- Stolcke, A., C. Barry, H. Franco, V.R.R. Gadde, M. Graciarena, M-Y. Hwang, K. Kirchhoff, A. Mandal, N. Morgan, X. Lei, T. Ng, M. Ostendorf, K. Sonmez, A.

- Venkataraman, D. Vergyri, W. Wang, J. Zheng and Q. Zhu. 2006. Recent innovations in speech-to-text transcription at SRI-ICSI-UW. *Audio, Speech and Language Processing, IEEE Transactions on*, 14(5): pp. 1729–1744.
- Stolcke, A., F. Grézl, M.-Y. Hwang, X. Lei, N. Morgan and D. Vergyri. 2006. Cross-domain and Cross-language Portability of Acoustic Features Estimated by Multilayer Perceptrons. *Proceedings of ICASSP'06*.
- Stolcke, A. 2002. SRILM - an extensible language modeling toolkit. *Proceedings of ICSLP*.
- Stolcke, A., H. Bratt, J. Butzberger, H. Franco, V.R.R. Gadde, M. Plauché, C. Richey, E. Shriberg, K. Sönmez, F. Weng and J. Zheng. 2000. The SRI March 2000 Hub-5 conversational speech transcription system. *Proceedings NIST Speech Transcription Workshop*, College Park, MD, May.
- Su, Y., F. Jelinek, S. Khudanpur. 2007. Large-Scale Random Forest Language Models For Speech Recognition, *Proceedings of ICASSP'07*.
- Tam, Y.C. and T. Schultz. 2008. Correlated bigram LSA for unsupervised LM adaptation. *Advances in Neural Information Processing Systems*.
- Tam, Y.C. and T. Schultz. 2005. Language model adaptation using variational Bayes inference. *Proceedings of Interspeech*.
- Tam, Y.C. and T. Schultz. 2006. Unsupervised language model adaptation using latent semantic marginals. *Proceedings of Interspeech*.
- Tam, Y.C. and T. Schultz. 2007a. Correlated latent semantic model for unsupervised language model adaptation. *Proceedings of ICASSP*.
- Tam, Y.C., I. Lane and T. Schultz. 2007b. Bilingual LSA-based LM adaptation for spoken language translation. *Proceedings of ACL*.
- Taskar, B., C. Guestrin and D. Koller. 2003. Maxmargin Markov networks. *Neural Information Processing Systems Conference (NIPS)*, Vancouver, B.C., Canada.
- Technology and Corpora for Speech to Speech Translation*, <http://www.tc-star.org>
- Tibrewala, S. and H. Hermansky. 1997. Multi-Stream Approach in Acoustic Modeling. *Proceedings of LVCSRHub5 Workshop*.
- Tillmann, C. and H. Ney. 2003. Word reordering and a dynamic programming beam search algorithm for statistical machine translation, *Computational Linguistics*.
- Tomalin, M. and P.C. Woodland. 2006. Discriminatively Trained Gaussian Mixture Models for Sentence Boundary Detection. *Proceedings of ICASSP*.
- Tree, F. 1995. The effects of false starts and repetitions on the processing of subsequent words in spontaneous speech. *Journal of Memory and Language*, 34, pp 709–738.

- Tsai, M.Y., F-C. Chou and L-S. Lee. 2007. Pronunciation modeling with reduced confusion for Mandarin Chinese using a three-stage framework. *IEEE Trans. on Audio, Speech, and Language Processing*, 15(2): pp. 661–675, Feb.
- Tseng, H, P. Chang, G. Andrew, D. Jurafsky and C. Manning. 2005. A conditional random field word segmenter. *Fourth SIGHAN Workshop on Chinese Language Processing*.
- Valente, F., J. Vepa, C. Plahl, C. Gollan, H. Hermansky and R. Schlüter. 2007. Hierarchical neural networks feature extraction for LVCSR system. *Interspeech*.
- Valente, F. and H. Hermansky. 2007 Hierarchical and parallel processing of modulation spectrum for ASR applications. *Proceedings of Interspeech 2007*.
- Valente, F. and H. Hermansky. 2006. Combination of Acoustic Classifiers based on Dempster-Shafer Theory of Evidence. *Proceedings of ICASSP 2007*.
- Valtchev, V., J.J. Odell, P.C. Woodland and S.J. Young. 1997. MMIE training of large vocabulary speech recognition systems. *Speech Communication*, 22: pp. 303–314.
- Vapnik, V. 1995. *The nature of statistical learning theory*. Springer.
- Venkataraman, A., A. Stolcke, W. Wang, D. Vergyri, V. R. Gadde and J. Zheng. 2004. An Efficient Repair Procedure for Quick Transcriptions. *Proceedings of ICSLP'04*.
- Venkataraman, A. and W. Wang. 2003. Techniques for effective vocabulary selection in domain specific speech recognition, *Proceedings of EUROSPEECH*, pp. 245–248.
- Venkataraman, A., A. Stolcke, W. Wang, D. Vergyri, V.R.R. Gadde and J. Zheng, 2000. An efficient repair procedure for quick transcriptions, *Proceedings ICSLP*, pp. 1961–1964, Oct.
- Vergyri, D., A. Mandal, W. Wang, A. Stolcke, J. Zheng, M. Graciarena, D. Rybach, C. Gollan, R. Schlüter, K. Kirchhoff, A. Faria and N. Morgan. 2008. Development of the SRI/Nightingale Arabic ASR system. *Proceedings Int. Conf. on Spoken Language Processing*, pp. 1437–1440, Brisbane, Australia, September.
- Vergyri, D., K. Kirchhoff, K. Duh and A. Stolcke. 2004. Morphology-based language modeling for Arabic speech recognition. *ICSLP*.
- Vergyri, D. and K. Kirchhoff. 2004. Automatic Diacritization of Arabic for Acoustic Modeling in Speech Recognition. *COLING Workshop on Arabic-script Based Languages*, pp. 66–73, Geneva.
- Vogel, S., Y. Zhang, F. Huang, A. Tribble, A. Venogupal, B. Zhao and A. Waibel. 2003. The CMU statistical translation system. *Proceedings MT Summit IX*, New Orleans, LA.
- Wallach, H.M. 2006. Topic Modeling: Beyond Bag-of-Words. *International Conference on Machine Learning*.
- Wang, L., M. J. F. Gales and P. C. Woodland. 2007. Unsupervised training for Mandarin broadcast news and conversation transcription. *Proceedings ICASSP*, Honolulu.

Wessel, F. and H. Ney. 2005. Unsupervised training of acoustic models for large vocabulary continuous speech recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 13(1): pp. 23–31.

Woodland, P.C. and D. Povey. 2000. Large Scale MMIE Training for Conversational Telephone Speech Recognition. *Proceedings of Speech Transcription Workshop*.

Xiang, B., L. Nguyen, X. Guo and D. Xu. 2005. The BBN Mandarin Broadcast News Transcription System. *Proceedings of Interspeech'05*, pp. 1649–1652.

Xiang, B., K. Nguyen, L. Nguyen, R. Schwartz and J. Makhoul. 2006. Morphological decomposition for arabic broadcast news transcription. *Proceedings of ICASSP*.

Xu, P., A. Emami and F. Jelinek. 2003. Training connectionist models for the structured language model. *Proceedings of EMNLP*.

Yu, K., M. J. F. Gales and P. C. Woodland. 2007. Unsupervised training with directed manual transcription for recognising Mandarin broadcast audio. *Proceedings of INTERSPEECH*, Antwerp.

Yu, H., Y. C. Tam, T. Schaaf, S. Stüker, Q. Jin, M. Noamany and T. Schultz. 2004. The ISL RT04 mandarin broadcast news evaluation system. *EARS Rich Transcription Workshop*, Palisades, NY, USA, November.

Zavaliagos, G. and T. Colthurst. 1998. Utilizing untranscribed training data to improve performance. *DARPA Broadcast News Transcription and Understanding Workshop*.

Zhan, P. and M., Westphal . 1997. Speaker Normalization Based on Frequency Warping. *Proceedings of ICASSP-97*, Vol. 2, pp. 1039–1041.

Zhang, B., S. Matsoukas and R. Schwartz. 2004. Long span features and minimum phoneme error heteroscedastic linear discriminant analysis. *Proceedings EARS RT-04 Workshop*, New York, September.

Zhang, B., S. Matsoukas and R. Schwartz. 2006. Discriminatively Trained Region Dependent Feature Transforms for Speech Recognition. *Proceedings ICASSP'06*.

Zheng, J. and A. Stolcke. 2005. Improved Discriminative Training Using Phone Lattices. *Proceedings of European Conf. on Speech Communication and Technology (EUROSPEECH'2005)*.

Zhang, J., R. Jin, Y. Yang and A. Hauptmann. 2003. Modified logistic regression: An approximation to SVM and its applications in large-scale text categorization. *International Conference on Machine Learning (ICML)*, Washington, DC, USA.

Zhang, J., J. Gao and M. Zhou. 2000. Extraction of Chinese compound words - an experimental study on a very large corpus. *The Second Chinese Language Processing Workshop attached to ACL2000*.

Zhang, R., G. Kikui, H. Yamamoto, T. Watanabe, F. Soong and W. K. Lo. 2004. A unified approach in speech-to-speech translation: Integrating features of speech

- recognition and machine translation. *Proceedings COLING*, Geneva, Switzerland, pp. 1168–1174.
- Zhang, Y. and S. Vogel. 2004. Measuring confidence intervals for the machine translation evaluation metrics. *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*.
- Zheng, J., O. Cetin, M-Y. Hwang, X. Lei, A. Stolcke and N. Morgan. 2007. Combining discriminative feature, transform, and model training for large vocabulary speech recognition, *Proceedings ICASSP*, pp. 633–636, Apr.
- Zheng, J. and A. Stolcke. 2005. Improved discriminative training using phone lattices, *Proceedings Eurospeech*, pp. 2125–2128, Sep.
- Zhou, B. and L. Besacier and Y. Gao. 2007. On efficient coupling of ASR and MT for speech translation. *ICASSP-07*.
- Zhu, Q., A. Stolcke, B. Y. Chen and N. Morgan. 2004. Incorporating Tandem/HATs MLP Features into SRI's Conversational Speech Recognition System. *Proceedings of DARPA Rich Transcription Workshop*.
- Zhu, Q., A. Stolcke, B. Y. Chen and N. Morgan. 2005. Using MLP features in SRI's conversational speech recognition system. *Interspeech'05*, pp. 2141–2144.
- Zolnay, A. 2006. Acoustic Feature Combination for Speech Recognition. *Ph.D. thesis, RWTH Aachen University, Aachen, Germany, August*.
- Zolnay, A., R. Schlüter and H. Ney. 2005. Acoustic feature combination for robust speech recognition. *Proceedings ICASSP*, volume 1, pp. 457–460, Philadelphia, PA, USA, March.

Chapter 4: Distillation

Editors: Radu Hans Florian, Joseph Olive Caitlin Christianson, and John McCary

4.1 Introduction

Author: Radu Hans Florian

The GALE distillation task was designed to synthesize information extracted from multiple source types and languages and distill them into coherent and complete English representations that could be directly used by analysts. This included detecting and presenting both redundancy and contradiction to make large amounts of information easier to digest. This chapter discusses the work done for the distillation task of GALE.

The distillation program has roots in and similarities to many fields in natural language processing, including question answering, information extraction, summarization, and machine translation. In the process of integrating these previously segregated fields, a large number of scientific explorations have been started and are presented in this chapter. Some of them focused on advancing the state of the art in information extraction, while others were directed at the interface between the components of the GALE program – speech recognition and machine translation. These technologies are combined and integrated to produce concise relevant and non-redundant responses to queries.

To focus the semantic search on a particular set of desired capabilities, a set of 17 interesting query templates was created. These templates covered a large spectrum of inquiries including the biography of a person, how a particular country reacted to a specific event, and where and when a person has been. In particular, systems had to identify all sub-sentential units relevant for a given semantic query, extract citations for each of them and group them into semantically equivalent sets by identifying and marking redundancy. The evaluation also measured system performance relative to a human performing the same task.

This part is organized into five chapters, each containing sections that address one of the GALE distillation objectives. 4.2 contains the design of the template task that was the focus of the distillation quantitative evaluation and was created to stress-test advances in the field (Section 4.6.1). 4.3 describes the generic architecture of a GALE Distillation System and two specific implementations of such systems, detailing the particular approaches devised by the AGILE and NIGHTINGALE teams. 4.4 describes general investigations into topics related to distillation, including identifying sentence relevance (Hakkani-Tür *et al.* 2010), comparing rule-based with statistical methods for sentence relevance detection (Section 4.4.3), system combination for Distillation Systems (Ma *et al.* 2010), using structural semantic information for language understanding (Freedman *et al.* 2010), and identifying contradictions in text (Section 4.4.6). 4.4 also contains a

detailed description of the natural language processing toolkit used by the AGILE team (Section 4.4.1). 4.5 describes investigations of the connection of distillation with the other components of the GALE program: using source-side named entity identification to improve translation (Ji *et al.* 2010), and using query-time available information to improve recognition of names in speech and translation (Section 4.5.1; Section 4.5.2). 4.6 discusses methods for evaluation of the technology.

4.2 Template-Based Query Development

4.2.1. The Templates

Authors: Connie Fournelle and Olga Babko-Malaya

GALE distillation templates provided a structured method for posing queries to the GALE engines. These templates eliminated the need for natural language analysis of queries, allowing researchers to focus on retrieval of relevant information.

The templates also provided a means to request a very specific set of information. Templates provide structure to queries that is much tighter than keyword search, enabling system users to ask open-ended questions without having to wade through a volume of irrelevant returned information in order to focus on a few desired pieces of information. Note that templates do, in fact, provide structure for a set of open-ended queries, rather than structuring a set of factoid-type questions.

In accordance with the program vision, all templates were written in English, and systems retrieved and presented relevant information in response to these templates in English.

We group the templates into three categories, as shown in the tables below: (Table 4.1) templates for events; (Table 4.2) templates for agents – i.e. persons, organizations, or countries; and (Table 4.3) templates involving relations between agents and events.

EVENT TEMPLATES		
	Template	Restrictions
1	List facts about [EVENT]	The activity date indicates the time interval of interest for the query. The location indicates the location of actions of interest for the query
2	What connections are there between [EVENT1/TOPIC1] and [EVENT2/TOPIC2]?	Location is not applicable
3	Describe attacks in [LOCATION] giving location (as specific as possible), date, and number of dead and injured	The activity date indicates the time interval of the attacks to be included in the response. The location indicates location of the attacks of interest for the query

Table 4.1: Event Templates

AGENT TEMPLATES		
	Template	Restrictions
4	Provide information on [PERSON]	Activity date and location are not applicable.
5	Provide information on [ORGANIZATION]	Activity date and location are not applicable.
6	Find statements made by or attributed to [PERSON] on [TOPIC(S)]	Location is not applicable. When an activity date is provided, responses should only include statements made during the specified time window.
7	Describe the relationship of [PERSON/ORG] to [PERSON/ORG]	If an activity date is provided, only include activities associated with the relationship that occurred within the specified time interval. When a location is provided, responses should only include evidence of the relationship that occurred in specified location.
8	Find acquaintances of [PERSON]	If an activity date is provided, response should be restricted to acquaintance ties that are explicitly declared to exist during the specified time period If a location is provided, responses should be restricted to acquaintances that are explicitly connected to the specific location
9	Produce a biography of [PERSON]	Activity date and location are not applicable.
10	Describe the actions of [PERSON] during [DATE] to [DATE]	The activity date indicates the time interval of interest for the query. The location indicates the location of actions of interest for the query
11	Describe the actions of [ORGANIZATION] during [DATE] to [DATE]	The activity date indicates the time interval of interest for the query. The location indicates the location of actions of interest for the query
12	Describe arrests of persons from [ORGANIZATION] and give their role in the organization	The activity date indicates time interval of the arrests of interest. The location is the location of the arrests
13	Where has [PERSON] been and when?	If the activity date is specified, only include locations whose known associated date falls within the time interval. Location is not applicable for this template

Table 4.2: Agent Templates

AGENT/EVENT TEMPLATES		
	Template	Restrictions
14	What [PEOPLE/ORGANIZATIONS/COUNTRIES] are involved in [EVENT] and what are their roles?	The activity date indicates the date of the event. The location indicates the location of the event.
15	Describe involvement of [PERSON/ORGANIZATION/COUNTRY] in [EVENT/TOPIC]	The activity date specifies the date of the event, or period of interest for the topic. When a location is provided, it specifies the location of the event or topic (ex: coal mining in West Virginia)
16	Describe the prosecution of [PERSON] for [CRIME]	The activity date indicates time frame for the prosecution. The location indicates the location of the prosecution.
17	How did [COUNTRY] react to [EVENT]?	The activity date indicates the date of the event. The location indicates the location of the event.

Table 4.3: Agent/Event Templates

4.2.2. Relevancy Determination

Authors: Connie Fournelle and Olga Babko-Malaya

4.2.2.1. Introduction

When developing responses to the templates, systems must determine whether information they encounter in the system is relevant, and if it is redundant with other information they have already selected. In the following section, we discuss how we scoped the requirements for each template via relevancy guidelines, and how consistently humans were able to determine the relevance of information in machine and human-generated responses to those specific queries written in these templates.

4.2.2.2. Relevance Guidelines

Relevancy guidelines included (1) description of information which should be viewed as relevant, and (2) a list of relevant “categories” for each template.

For each template, we provided general information to help an annotator or system developer determine whether information was relevant to the query. To further characterize relevant information within the guidelines, we also provided information categories to indicate the reason for information relevance. At least one category had to apply for the information to be relevant. Categories existed for illustrative purposes only; training data did not include category annotations, and distillation engines were not asked to provide category labels in their output.

4.2.2.3. Relevancy Guideline Examples

To illustrate the task specification associated with each template, we provide the following example, drawn directly from the GALE program Relevancy Guidelines:

Instructions for Template 1: LIST FACTS ABOUT [event]

Relevant information focuses on the event and the persons, places, and activities directly associated with the event. Relevant information also includes subevents, causes, goals, and precursor or preparation events.

For persons associated with the event: relevant information includes their role and the reason for their involvement in the event, as well as, their involvement in promoting, funding, or planning for the event. Information about a person must be directly related to the event if it is to be relevant. Involvement in the event may be confirmed or suspected.

Direct reactions, direct consequences, and the significance of the event are also relevant.

For locations associated with events: only information that directly pertains to the event is relevant.

Activity date: *indicates date of the event*

Location: *indicates the location of the event*

Categories: *time, location, cause/intention/planning, participant, subevent/execution/manner, consequence/reaction/significance*

Example Responses for Template 1:

LIST FACTS ABOUT [EVENT]: [Vice President Dick Cheney's shooting of Harry Whittington]

Relevant:

Subevent/execution/manner: Whittington was shot with pellets

Location: on a private Texas ranch

Manner: Shooting was accidental

Reaction: Shooting has generated much talk

Participant: Whittington is Cheney's hunting partner

Effect: A pellet lodged near Whittington's heart caused his heart attack

Irrelevant:

Whittington is a lawyer

Whittington is a prominent Republican

Whittington is 78 years old

4.2.2.4. Query Responses

Distillers, whether they are humans or machines, produce (English-only) query responses consisting of snippets of text, which may include direct quotations from source text, translations of foreign materials, and summaries of these materials. Answers provided by distillers were at the sub-document, and even sub-sentence level, while still maintaining full traceability to the original source document. Distillers were instructed to provide only relevant information, which the evaluators subsequently parsed into atomic pieces of information contained in the responses. Scores for distillation engine were ultimately based on generating precision and recall scores for each distiller, and comparing those results to human performance.

Evaluators only parsed relevant information; any irrelevant material returned by a distillation engine ultimately contributed towards the false detection score for the system. Consequently, the ability of annotators to understand and recognize which information was relevant was critical. Annotators further had the flexibility to assign partial relevance scores, or “degrees of relevance”, to atomic answers they created. While evaluating system responses, annotators were asked to categorize each relevant atomic fact according to the categories provided in the relevancy guidelines, which helped them in making decisions about relevancy of each created atomic fact. This approach, as we will show, has proven to produce consistent annotations with high inter-annotator agreement scores.

4.2.2.5. Inter-Annotator Agreement

We measure inter-annotator agreement for each template by measuring the *Relevancy Agreement*. This measure reflects yes/no judgments on a snippet level. Every annotated snippet that has at least one atomic fact is considered relevant. Snippets from which no atomic facts were created are irrelevant. *Relevancy Agreement* counts the percentage of snippets which both annotators judged as relevant or irrelevant. The inter-annotator agreement for Phase 2 for the different templates was between 76% and 100% with a mean of 89%.

4.2.2.6. Surprise Template

One potential concern with the GALE model of template-based queries was that work involved in adapting to a new query template might be cost prohibitive for new customers or new system users. To mitigate this concern, teams were asked to provide responses for one additional “surprise” template. Teams were given a description of the template and provided with 24 hours to adapt their systems to this new template. The idea behind the surprise template was to test the systems’ ability to adapt to a new template within 24 hours, in case field users were unable to procure desired information from existing templates. One Surprise Template was:

IDENTIFY ADDITIONAL NAMES/SPELLINGS/ALIASES FOR
[PERSON/ORGANIZATION /LOCATION] AND DATE OF ANY CHANGES TO NAME IF
KNOWN

Instructions for Surprise Template:

Responses should include various spellings, aliases, nicknames, or name changes for the same person, organization, or geographic location IF the multiple spellings/aliases/name changes are noted in the same document. [We are not interested in lists of “equivalent terms” from a gazetteer. All information must come from the corpora text.]

Examples of relevant responses include sentences that explicitly state that name variants refer to the same entity, such as entities followed by phrases like “also known as,” “formerly known as,” or “will soon be called.” Also relevant are constructions where entities are followed by another name in parentheses - as in *Jerusalem (Al-Quds)* - or followed by a nickname in quotations - as in *Chevrolet, “Chevy” or Stephen “The Rifleman” Flenmi.*

Different spellings of the name which occur in the same document are also relevant.

Teams are not required to co-reference within documents to provide the additional names if world knowledge is needed to infer that the name variants refer to the same entity - as in *Her soaring I happen To Like New York is a lusty, life-affirming tribute to the Big Apple.* Such snippets will not count as relevant, however, there will be no penalty for providing this information.

Do not include responses which restate a part of the same name. For instance, if a sentence says *“Bill Clinton was President of the United States, but Clinton’s tenure was tumultuous,”* do not list “Clinton” as a variant of “Bill Clinton.”

For responses which include a name change, also relevant is a date of change (past, present, or future). Dates pointing to when a person/place/organization were called by its old or new name are relevant, even if the exact date of change is not given.

If an organization changes its name, include any change/non-change to its stock ticker if available.

Responses should not include reasons for name change, expected impact of changes, or possible consequences of the change. Also irrelevant are statements about the change, impact, or cause of the change.

As this example shows, the specifications for new templates were, in fact, reasonable guidelines that an interested party could provide, and since teams could adapt their systems within a 24 hour period, the model of template-based query was operationally tractable.

4.3 Architecture and Implementation of a Distillation System

4.3.1. Architecture of a Generic GALE Distillation System

Authors: Radu Florian, Connie Fournelle, Kathy McKeown, and Ralph Weischedel

4.3.1.1. Introduction

This section presents the generic architecture of a GALE Distillation System. The usual process for a GALE Distillation System consists of two steps, the first being processing of data into text, representing it in a form that is appropriate for searching, and the second consisting of a runtime module, which performs the actual query.

The following two sections define the two phases of a Distillation System.

4.3.1.2. The Preprocessing Stage

The preprocessing phase usually consists of:

- Speech recognition, applied to audio sources
- Processing in the source language (sentence segmentation, etc.)
- Machine translation, applied to texts in a foreign language (e.g., Arabic, Chinese)
- Information extraction applied to both foreign language and English text (such as sentence segmentation, parsing, mention detection, co-reference resolution, relation and event detection and recognition, semantic role label detection, etc., see Section 4.4.1) presents an extensive list of natural language processing that is typical of this phase.

Figure 4.1 presents a typical data processing flow. Speech input – e.g., broadcast news, broadcast conversations, telephone conversations, etc. – is passed through an

automatic speech recognizer, for both the English and the non-English content, resulting in text documents, which are added to the collection of documents in text format, which usually consists of newswire, weblog, and newsgroups. These text documents are then processed by extracting semantic and syntactic information, extracting named and non-named entities, resolving co-reference, identifying relations and events, either in the source language of the document and translating the extracted information into English, or by translating the foreign source document into English and performing the extraction in English, or possibly both. This processed document collection now represents a rich and enhanced pool in which the systems will search for answers to the user queries.

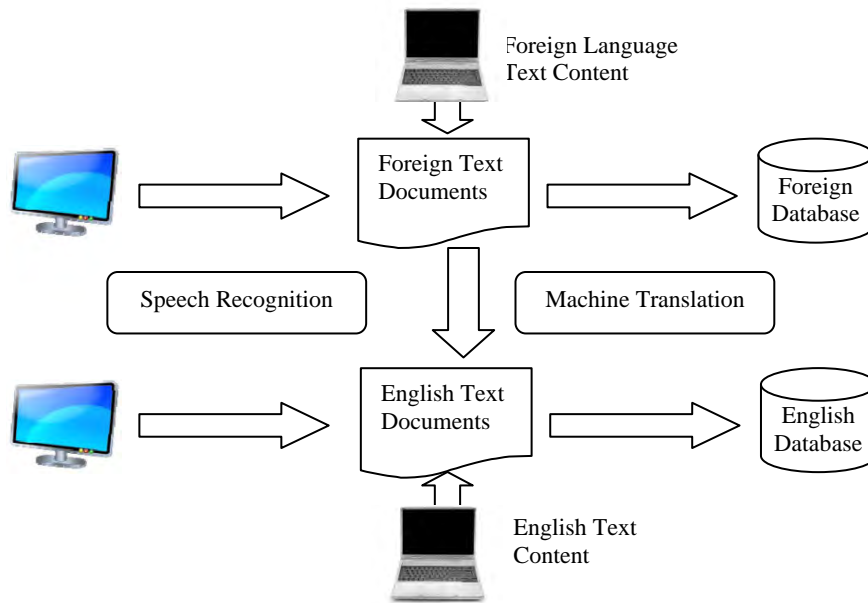


Figure 4.1: Typical Data Preprocessing

It is notable here that the lines between the collections and processing do not necessarily have to be as clear as they are shown in

Figure 4.1. As it will be shown in the next section and in Section 4.5.3 information gathered by identifying named entities in the source language can be used to perform a more faithful translation of the text.

4.3.1.3. The Runtime Stage

The runtime phase of a Distillation System consists of processing a stream of user queries. Figure 4.2 depicts such a typical processing sequence – the query is processed by the template processing engine by identifying the arguments, their types, and relations among them, running a document search in the processed information database created in the preprocessing phase, then extracting snippets – basically semantically coherent sentential or subsentential units – from the found documents, then identifying and

marking redundancy among them and extracting and grouping citation information – the source text support the snippet.

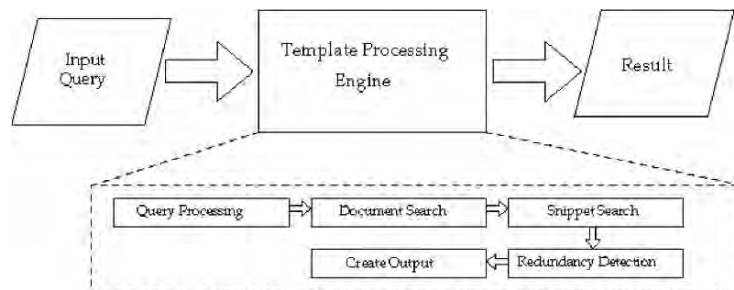


Figure 4.2: Runtime Processing

Below is a short description of each stage:

- Query preprocessing consists of identifying the arguments of the query, mostly in refining the input into smaller components (such as breaking the titles from the name, e.g., “[president] [George Bush]”), linking the named arguments with a cross-document database (identifying which George Bush the query is about), identifying relations in the arguments (“George Bush” has the occupation “president”).
- The separation between the document search and snippet search is not theoretically necessary, but it is practically needed (and sometimes leads to increased performance), as performing advanced processing on all the documents is not feasible for large document collections. During the document search phase, one would create a search query that is based mostly on lexical atoms (the words in the arguments), and other features that are discovered during the query preprocessing phase. The output from this phase is a relatively small number of documents.
- During the snippet search phase, the documents identified at the previous stage are investigated for sentences / sub-sentential units that match the given query. One can afford to do deeper processing here, using both syntactic and semantic information, given the relative small number of documents. This is the core of the Distillation System, and this is the place where the systems will tend to differ the most. The output from this stage is a list of snippets that are relevant to the query.
- The resulting snippets are then checked for redundancy: snippets are scheduled to be output in some order (very likely in the order of their likelihood to be relevant), and snippets that are found to completely subsume previous snippets are marked as citations for those snippets, and any redundant part of the snippets is marked correspondingly.
- Finally, a number of the resulting snippets are output, taking into account the desired output size for the particular query and general guidelines for the output

(output size, format, etc.). Redundant parts in any snippet relative to previously output snippets are marked and output correspondingly.

As with the preprocessing phase, systems can make use of the input query to refine the search process. An example can be found in Section 4.5.2, where the query arguments are used to perform acoustic adaptation for a better search through the audio documents. Similarly, systems can use this information, and cross-lingual information retrieval CLIR techniques, to do a better job at doing the document search (and possibly retranslation) of material in foreign language – the main idea being to use query adaptation techniques for speech recognition and machine translation.

The following two sections present examples of implementations of GALE Distillation Systems: the AGILE system (Section 4.3.2) and the NIGHTINGALE System (Section 4.3.3).

The architecture of a GALE field system is similar to a generic laboratory system in the preprocessing phase, but differs significantly in the runtime processing. Here are some examples of differences:

- The system has to be interactive, visualizing the results, helping analysts to generate reports, allow the users to also perform other tasks (such as regular textual search)
- The system has to be responsive, which might impose limitations of not using compute intensive methods (such as speech re-decoding)
- The system has to allow the user to refine/adjust queries

4.3.2. Distillation at BBN – The AGILE System

Authors: Elizabeth Boschee, Marjorie Freedman, Roger Bock, John Graettinger, and Ralph Weischedel

4.3.2.1. Introduction

Distillation at BBN involves two primary stages of processing: background processing and live distillation. Background processing includes speech recognition, machine translation, running the Statistical Entity and Relation Information Finding (SERIF) information extraction engine, creating word and extraction based indices and identifying aliases and misspellings for the named entities in the corpus. Live distillation includes four stages: query understanding, relevant document retrieval, answer selection, and response generation.

4.3.2.2. BBN Background Processing

Every document in the corpus is processed and analyzed before any questions are asked. The results of this analysis are stored in a database and are immediately accessible to the system during live distillation.

- Figure 4.3 illustrates BBN’s background processing. Speech recognition systems convert speech data into text. This text is then segmented in stories. Arabic and Chinese documents are translated by one or more machine translation systems. BBN’s Distillation System works over multiple translations (or transcriptions) of the same text. The resulting Distillation Corpus is processed by BBN’s core NLP engine, SERIF. SERIF is described in detail in Section 4.4.1. Highlights of SERIF’s document analysis include: Entity extraction: For each entity (person, organization, place, etc.) mentioned in the document, the system identifies all named references to that entity, as well as, all descriptions and pronominal references.
- Relationship extraction: The system detects and classifies relationships between entities, e.g., “*Bill Gates is the CEO of Microsoft*” → Employment (Bill Gates, Microsoft).
- Event extraction: The system detects and classifies events and their arguments, e.g., “*The bombing in Baghdad injured three Americans*” → Attack (Victim: three Americans, Location: Baghdad)
- Parsing and proposition extraction: The system generates a syntactic parse for each sentence and from that parse extracts a set of predicate-argument propositions (i.e. who did what to whom) that together form a normalized representation of the parse structure.

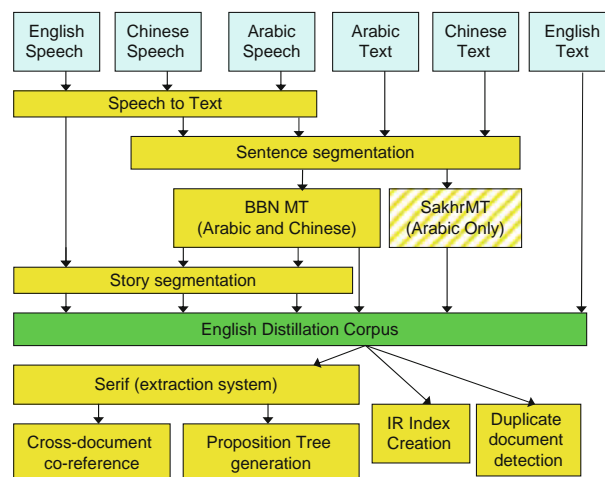


Figure 4.3: BBN Background Processing

For documents that originated in a language other than English, this analysis is performed both for the source language document and for each machine translation that is provided.

SERIF analyzes each document independently, facilitating the addition of new documents to the corpus as they become available. Additionally, BBN produces a *tf/idf* index of the corpus (to enable document retrieval) and a cross-document co-reference database of the named entities mentioned in the corpus documents.

The cross-document co-reference database is generated using algorithms that handle variation in names across news sources (and languages), including aliases, misspellings,

transliteration variations, and abbreviations. The cross document co-reference system clusters names together based on document information generated by SERIF, information provided by external resources (e.g., known alias lists, geographical databases, Wikipedia), and other algorithms designed to handle aliases and spelling variation (e.g., Levenshtein's edit distance). A sampling from a cluster generated by this system is: al Qa'eda, al Qa'ida, ll Qaeda, al Quaida, al Qaedas, al Qaeda Group of Jihad, al Qaeda movement, etc. The cross document co-reference database is designed to account for spelling variation and aliases, but not to disambiguate between two entities that share a name.

4.3.2.3. BBN Real-Time Distillation

Real-time question-answering begins when the user asks a query. As illustrated in Figure 4.4, there are four stages to BBN's real-time processing: query interpretation, document retrieval, answer finding, and redundancy removal / response generation. We describe each of these in detail here.

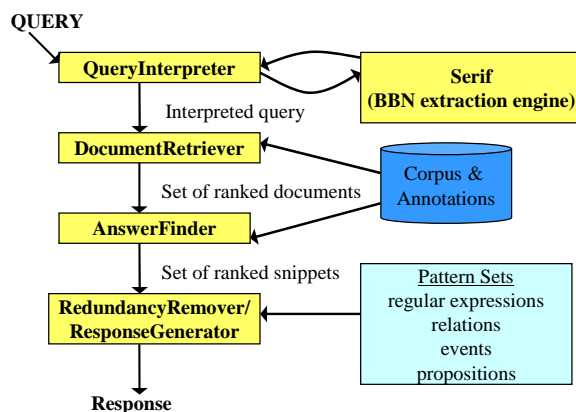


Figure 4.4: BBN's Live Real-time Distillation

4.3.2.4. Query Interpretation

In the first stage, BBN uses SERIF to analyze the query provided by the user. It also translates the query into both Arabic and Chinese and uses SERIF to analyze the translated queries as well.

For templated queries, only the query slots are analyzed. For instance, for the query "Provide a biography of [Kenyan president Mwai Kibaki]", the system will analyze only "Kenyan president Mwai Kibaki". In this instance, it will identify "Kenyan" and "Mwai Kibaki" as names and detect that "Mwai Kibaki" is the name of the person whose biography is sought.

BBN's system is also capable of responding to non-templated questions. For free text (non-templated) queries, the system will use the SERIF analysis to decide what answer strategy is appropriate for this query. For instance:

- “Tell me about Mwai Kibaki” → find-biography(“Mwai Kibaki”)
- “Where was Bill Clinton born?” → find-location(“Bill Clinton was born”)
- “Who won the 1991 World Series?” → find (“X won the 1991 World Series”)

Further SERIF analysis will provide information beyond the selection of the answer strategy. For instance, for “Where was Bill Clinton born?”, the system will decide not only that this is a request for a geographical location, but also that the event in question can be normalized as Birth (Bill Clinton), enabling the next stage to easily select all documents where that event has been seen.

4.3.2.5. Document Retrieval

In the second stage, the BBN Distillation System retrieves a set of documents that it judges most likely to contain answers to the user’s query. There are two primary methods of document retrieval: a tf/idf index and an extraction database.

The extraction database serves as the primary method of document retrieval, with the tf/idf index being used primarily for re-ranking or when there is no extraction information available for the query (this will happen when the query contains no named entities, relationships, or events, e.g., “List facts about [the mad cow disease outbreak]”).

Each type of query has a document retrieval strategy associated with it, made up of one or more sub-strategies. Available sub-strategies include:

1. Find documents containing name X.
2. Find documents containing name X and name Y.
3. Find documents containing a relationship between name X and name Y.
4. Find documents containing an event of type E with an argument of name X.
5. Find documents containing both name X and an event with type E.

These searches are performed over all translations of the corpus documents, as well as, the source-language versions of the corpus documents, using the translated version of the query. Document- and entity-level statistics are used to rank the results returned by these sub-strategies. There are also full text and fuzzy text search sub-strategies available to the system.

The combination of these sub-strategies results in a ranked list of documents. The system then uses this list of documents to constrain a tf/idf search incorporating all of the words from the query. The scores from the tf/idf query are normalized and combined with the normalized scores from the database queries.

For instance, we consider the query “*List facts about [the contested election of Mwai Kibaki]*”. During the query interpretation stage, BBN’s Distillation System will identify that this query contains an Election event with Mwai Kibaki as an argument. During the document retrieval stage, the system will therefore search first for documents containing Election events with Mwai Kibaki as an argument (Strategy 4). If that does not return enough documents, then it will search for documents containing Election events and the name Mwai Kibaki (Strategy 5). If this still returns only a few documents, the system will

then back-off to searching for all documents containing Mwai Kibaki (Strategy 1). The system will then run a tfi/df query (using the words contest election Mwai Kibaki), constrained to include only those documents that the database found. The scores of the tf/idf search will be combined with the scores from the database search and a ranked list of documents provided to the next stage.

As mentioned previously, BBN's preprocessing produces multiple translations of the same source text. During document retrieval, additional translations are included to increase the likelihood that relevant snippets are found in the text (the Response Generation stage ensures that we do not produce overlapping answers from the same source document).

4.3.2.6. Answer Finding

In the third stage, BBN's Distillation System uses patterns to find answers to a query in the set of documents returned by document retrieval. To do this, the system must first match the entities or topics in the query with the entities and phrases in the document. Entity matching is performed using a combination of the within-document co-reference information (linking names to descriptions and pronouns) and the cross-document co-reference database (linking names to other names). So, the system is able to match "Find statements by [Bill Clinton]" to "he said..." Topic matching is performed using the proposition trees (or text graphs) described in Section 4.4.4.4. This matching strategy allows the system to detect the similarity between phrases that are semantically similar but do not share surface structure or key lexical items. For example identifying the similarity between "*Clinton met the Kenyan president in July 2005*" and "*the meeting between Clinton and Kibaki happened in 2005*"

Once the entities and topics have been found in a document, the system uses query-specific patterns to identify sentences that contain answers. The query-specific patterns use extraction structures (e.g., relations and events), syntactic structures (propositions), and regular expressions. For instance, for the query "*Provide a biography of Bill Clinton,*" one pattern might be [find all snippets that express a "family/social" relation with Bill Clinton as an argument]. This pattern would select both "*Chelsea, daughter of former president Bill Clinton*", and "*Clinton's friend Vernon Jordan.*"

This stage generates a ranked list of answer "snippets" (text extracted from corpus documents), which are delivered to the fourth and final stage of the live question-answering process.

After answer finding, as described in Section 4.4.5, a statistical process merges and re-ranks the response snippets from BBN and PowerAnswer.

4.3.2.7. Redundancy Removal and Response Generation

The goal of this stage is to identify and remove redundant information from the set of answer snippets identified by the previous stage. To do this the system must identify all phrases that overlap in meaning, for instance, "*Abu Abbas was arrested in Iraq*" and "*his capture in Iraq*". For this process, the system relies primarily on "proposition nodes" as the atomic unit of redundancy and output. These are best described as parse nodes, normalized with regard to predicate-argument structure, thus, they are the system's

closest approximation for a “concept” or nugget in the text. The system uses a bottom-up algorithm for finding directional similarities between nodes, using exact match, synonyms, syntactic structural comparisons, and cross-document co-reference information. In practice, redundancy between a pair of snippets, or nodes, is rarely a true or false quantity, so we assign weights to the similarity between any two phrases.

The system then uses these weights to decide which snippets should be included in the response and which should be reported only as supporting information. The system also decides at what point the response is more likely to lead to spurious irrelevant information than to add to new, relevant information being presented, and prunes the response accordingly, using a number of features such as the similarity of a given snippet to all previous snippets and the level of confidence that system has in the snippet.

At the end of this fourth stage, the complete response — a ranked list of relevant snippets, as well as, sets of additional redundant snippets reported as supporting information — is returned to the user.

4.3.3. Distillation at SRI - The NIGHTINGALE System

Authors: Kathleen McKeown, James Allan, Ralph Grishman, Dilek Hakkani-Tür, Julia Hirschberg, and Gokhan Tür

4.3.3.1. Introduction

SRI’s NIGHTINGALE Distillation Engine features information retrieval tuned by templates from a set of richly annotated documents, a suite of response generators also tuned for the different templates, and a redundancy detection component that operates below the sentence level.

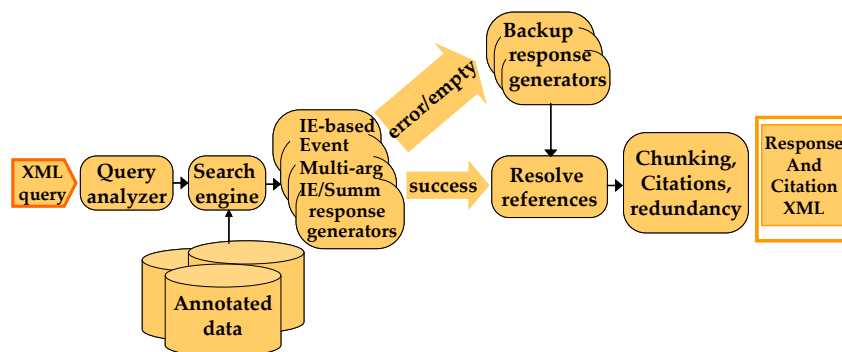


Figure 4.5: NIGHTINGALE Distillation Architecture

An overview of the system architecture is shown in Figure 4.5. It shows that the first step in processing is to parse the input XML query, retrieving template type and query arguments. The UMASS INDRI information retrieval engine (see Section 4.3.3.3) is called next with both the arguments and template as input. It searches through the corpus collection, which has been annotated with a variety of linguistic information, including parses, POS tags, and named entities. The relevant documents are retrieved and the

template specific response generator is invoked to extract relevant sentences from the documents.

The NIGHTINGALE System had multiple response generators, some of which handle more than one template type. In some cases, multiple response generators were used for the same template and the responses were merged to produce the final answer. If the primary response generator failed to return a response, the back-up response generator was called. We used a generic, query-focused summarization system as back-up, which has not been tuned for any of the template types. The generated response was then passed through redundancy detection, a system which uses a graph based method to determine which sentences should be included in the response and which should be included only as redundant sentences. The redundancy detector also retrieves the citations. Finally, the name rewriting system was called to include full names and full temporal references for every pronoun and partial names.

One feature of NIGHTINGALE distillation was its ability to handle noise in the input documents. It included a facility to generate name variations in order to retrieve more relevant information when the query contained a name as an argument. It also exploited the speech features that were available in annotation to better access relevant information in speech documents. In the following sections, we describe the different components of the system.

4.3.3.2. Annotated Documents

An extensive set of annotations were added to the entire distillation corpus for use by the response generators. These annotations generally followed the format and content of the ACE [Automatic Content Extraction] 2005 guidelines. These included:

- named entities (including people, organizations, and locations)
- noun phrases referring to people, organizations, locations, and co-reference relations among these NPs
- date and time expressions, including normalized dates or date ranges when computable
- instances of 18 semantic relations, such as affiliation with an organization
- instances of 33 types of events

Several of the ACE event types correspond closely to events sought by the GALE query templates, such as prosecutions and attacks. These annotations were generated for English sources. For Arabic and Chinese sources, annotations were generated on the MT output. In addition, for Chinese, a second set of annotations were produced directly from the source language texts and then mapped to the MT output using the Chinese-English alignment information produced by the MT system. Furthermore, for English text sources we produced parse trees and predicate argument structures in GLARF [Grammatical Logical Argument Representation Framework].

4.3.3.3. The Indri Search Engine¹

The information retrieval part of the system was used to search for documents likely to contain answers to the questions. All documents were indexed using the Indri IR system from the UMass and CMU Lemur toolkit, a state-of-the-art retrieval system that combines language modeling probability estimates and field support wrapped in a highly expressive query language.² Mandarin and Arabic documents were indexed both in the source language and in translation; audio material was indexed after being transcribed and translated. In addition to capturing the text of the documents, all annotations were indexed so that they could be queried directly - e.g., finding references to “Austin” used as a person's name. The system also included an estimate of document quality that was used to decrease the chance that poor-quality documents would be retrieved when more readable - and therefore parse able - relevant material was available.

GALE template queries were mapped to Indri queries that were tuned on training data: each template, therefore, generated a query that combined keywords, fields, proximity operators, and weights differently. For example, a template that required information about a person would prefer documents where the name was actually tagged as a name, but would also consider the name string if it wasn't tagged, and would ultimately consider matching only parts of the name. Cross document co-reference information was incorporated if appropriate to support alternate forms of the name, though that ended up not being helpful in the evaluation.

Finally, by using Indri as a search engine, it possible for other members of the NIGHTINGALE project to issue queries separately from the GALE templates. This capability meant that they could do corpus-based processing of the parts of the query to better adapt to its peculiarities: if a location occurs thousands of times in a corpus, there may be little value in backing off to a more general location.

4.3.3.4. Response Generators

NIGHTINGALE used a variety of different approaches to response generation. These included an approach based on detection of event sub-parts, an approach involving interleaved IR, information extraction (IE), and summarization, an approach for multi-argument queries (e.g., Template 6), a machine learning based approach, a combined IE and summarization based approach, and an IE based approach. In this section, we provide some examples of the different approaches that were taken.

The event detection based approach was used for Templates 1 and 10. The NIGHTINGALE System uses a form of unsupervised learning over documents returned by the Indri information retrieval engine to create a shallow semantic network representation of a document collection corresponding to the event in question. This semantic network is used to identify important event constituents (i.e., statistically significant atomic relations, Filatova and Hatzivassiloglou, 2003) and the sentences that best cover these constituents, and thus, should be included into the final answer.

¹ <http://www.lemurproject.org/indri/>

² <http://www.lemurproject.org>

Interleaving of IR, IE and focused summarization was used for Template 8. We interleave generation, using IR in high precision mode in the first stage to return a small number of documents that are highly likely to be relevant. Information extraction of entities and events within these documents is then used to pinpoint highly relevant sentences and associated words, including name variants are selected to revise the query for a second pass of retrieval, improving recall. As part of this process, we approximate the relevant context by measuring the proximity of the target name in the query and extracted events. We employ an IE system trained for the tasks of the 2005 Automatic Content Extraction evaluation, which include entity and event extraction. ACE defines a number of general event types, including *Justice* events, which cover indictments, accusations, arrests, trials, and sentencing. The union of all these specific categories gives us many of the salient events in a criminal justice case from beginning to end. The program uses the events, as well as, the entities, to help identify the passages that respond to the question.

Query-focused summarization was used for Templates 3, 4, 13 and 14. This approach takes a person or organization as input. It filters all sentences which mentioned the entity or organization and extracts these sentences. The sentences are then clustered and a representative sentence from each cluster is chosen for the response. The clusters are rank-ordered by their relevance to the query, and also, by how much overlap there is with sentences already included in the response.

We use a combination of IE and query-focused summarization for Template 12, which produces a biography for an individual. IE biography patterns are learned using a combination of search on Wikipedia and Google. Seed tuples for fields such as date of birth, place of birth are extracted from formatted tables in Wikipedia. Sentences containing these tuples are identified in the accompanying text and the patterns corresponding to their usage are constructed. These patterns are used to identify additional sentences from Google. For fields which are not likely to have any reliable patterns (e.g., religion of an individual), we use two language models to distinguish between sentences that are likely to be biographical sentences and those that are unlikely to be biographical. A biographical language model is trained from Wikipedia biography articles and a non-biographical language model is trained from TDT news articles. Sentences extracted from relevant documents containing the query name are kept for the answer if they are closer to the biographical model than the non-biographical model.

Some of the response generators rely primarily on information extraction. These include Templates 15, 16 and 17, which are looking for information about arrests, attacks and locations. For the first of these, we use annotated events in the corpus to identify sentences containing arrests and attacks. For the last of these, we use named entity identification to locate the queried person or organization and extract the location that occurs in the same sentence.

Finally, another method that we use for response generation is based on supervised learning. We used the training data provided by the Linguistic Data Consortium (LDC) of paired queries and responses and learned features that could help to identify relevant sentences. We employ a data-driven statistical method for sentence extraction in information distillation, and treat the problem as a binary classification task, where each sentence is classified as relevant (positive), or not relevant (negative). For extracting

classification features, all names in each candidate sentence and query are extended with possible variants. A set of syntactic and semantic features is computed from these sentences, and a graph structure, that is used for feature extraction, is formed (Levit, *et al.* 2006). The syntactic features include part-of-speech tags and syntactic parse trees, and the semantic features include information extraction annotations, such as named entities, and semantic role labels. The names matching the ones in the query are also marked as so. *N*-gram and inclusion features are extracted from these graphs, and enhanced with various similarity based features between the sentence and the query.

4.3.3.5. Use of Speech Features

We conducted experiments to determine the best unit to choose for response generation, considering hypothesized sentences, automatically identified intonational phrases, and pause-based chunks (separated by 250ms and by 500ms of silence). We extracted features for training automatically from the speech signal associated with the appropriate chunks and also used hypothesized speaker turn and speaker identity information provided by the GALE SRI annotation team to train classifiers for each segmentation type. Comparison with hand-written summaries showed that intonational phrase-based summarization yields the best performance in terms of F-measure, with a score of 0.516, compared to sentences (0.434), 250ms pause-delimited segments (0.432) and 500ms pause-delimited segments (0.381). We used these findings to improve the response generation engine for Template 5 queries.

We also evaluated alternative input segmentations for news story boundary detection in: English, Mandarin and Arabic. Our results suggest that the best performance can be achieved by using 250ms pause-based segmentation or sentence boundaries determined using a very low confidence score threshold.

We integrated prosodic and acoustic information and disfluency detection into the distillation process for templates other than Template 5. We used this information to identify speaker roles in Broadcast News (anchor, reporter, other), which we also use to improve response generation.

4.3.3.6. Redundancy Detection

Practical attempts at reducing redundancy in the output of these systems, however, usually involve clustering the sentences of the generated output, picking a representative sentence from each cluster and discarding the rest. Although redundant information can be removed in this manner, clustering approaches tuned for coarse matches will also remove non-redundant information, whereas clustering approaches tuned for near-exact matches will end up removing very little repeated information. This is simply a consequence of the fact that information can, and commonly does, exist at the subsentential level and that clusters of sentences don't necessarily correspond to clusters of information.

In NIGHTINGALE, we use a novel graph-based representation to detect redundancy within documents given any high performance algorithm for identifying similarities between sentences. We identify redundancy at the sub-sentential level between the sentences of a document through alignment and use this to build a bipartite graph which

enables us to keep track of redundant information across sentences. Common information between pairs of sentences, detected with the alignment algorithm, can be extrapolated to document-wide units of information using the graph structure. Individual sentences that are encompassed by the information in the rest of the document can then be identified and removed efficiently by using a well known greedy algorithm adapted for this representation.

4.4 Enabling Technology Breakthroughs to Improve Distillation Capabilities

4.4.1. SERIF Language Processing — Effective Trainable Language Understanding

Authors: Lance Ramshaw, Elizabeth Boschee, Marjorie Freedman, Jessica MacBride, Ralph Weischedel, and Alex Zamanian

4.4.1.1. Introduction

For a decade, our vision has been a trainable, language-independent understanding system that would:

- handle multiple genres, such as newswire, broadcast news, blogs, email, and meeting and conversation transcripts
- be portable to other languages and other domains, given training data
- be robust to noisy input, e.g., speech recognition, OCR, and machine translation

For BBN's Distillation System, SERIF serves as the core NLP engine which extracts the features used for document retrieval, answer selection, and redundancy detection. SERIF (Statistical Entity and Relation Information Finder) is a trainable, language-independent understanding system that produces rich linguistic analyses (including full parses, *who did what to whom*, and co-reference). A propositional representation derived from the parses provides semantic features used in co-reference, relation, and event models.

The names, relations, and events extracted by SERIF supplement word-based document retrieval. The propositions, relations, and events serve as features that identify relevant snippets in a document. SERIF's entity co-reference allows the system to identify snippets even when a relevant entity is referred to with a pronoun or description. Section 4.4.1.2 describes these uses in more detail. In Section 4.4.1.3.5 we describe how we use a distance metric defined over SERIF's propositions to find snippets that are similar to a topic, or event specified in a query.

Historically, the initial motivation for SERIF was the information extraction task as defined in NIST's Automatic Content Extraction (ACE) evaluations.³ Rather than using handcrafted finite state patterns based on shallow parsing, patterns that identify only base noun phrases and verb groups (a common approach to information extraction systems 10 years ago), our design choice was a full parser, trained on a Treebank (Marcus *et al.* 1993). Rather than mapping directly to domain-specific extraction targets (specific classes of entities, relations, and events), our design was to map text to a propositional structure. Since the work started prior to the release of the first PropBank for English (Palmer *et al.* 2005) and since our goal was to capture predicate argument structure for more than verbs, we created a more inclusive propositional representation. Linguistic co-reference was always included, at first rule-based, then using trained statistical models.

The system maps from propositional structure to an application-specific representation, e.g., the entities, relations, and events of an ACE evaluation, or to an application-specific ontology for a reasoning system.

SERIF has been extensively applied and repeatedly tested in ACE (information extraction), GALE (query-response distillation and the *who, what, when, where, and why* of distillation), and AQUAINT (question answering). Its multilingual capability has been tested and evaluated in English, Arabic, and Chinese in ACE, GALE, and AQUAINT.

4.4.1.2. Designed for Multiple Contexts

SERIF's design has been driven by a diverse set of application goals well beyond information extraction from clean text:

- In AQUAINT and GALE, SERIF analyses of both queries and texts are matched at the entity, proposition, and relation levels to identify segments responsive to a query and to reduce redundancy in complex answers.
- To improve MT performance, SERIF output has been used to derive dependency trees for syntax-directed translation (see Shen *et al.* String-to-Tree Translation Models, in 2.4.5).
- SERIF has been used to supply the knowledge for a system that learns to read relations and concepts (Boschee *et al.* 2008).
- A windowing version was used to process speech recognition input, where sentence boundaries must be inferred (Ramshaw *et al.* 2001).
- SERIF was applied to noisy OCR output.
- SERIF can produce alternative parses to focus on distinctions that will produce usefully different interpretations.

4.4.1.3. SERIF System Description

The system's overall structure is shown in Figure 4.6. The initial stages of processing recognize relevant document metadata and break text into sentences. Each sentence in

³ ACE results site. NIST: <http://www.itl.nist.gov/iad/mig//tests/ace/>

turn is then processed through various interpretation models. Once sentence processing has been completed, the document-level models (like co-reference models) are run.

Figure 4.7 shows sentence level processing. The sentence string is first tokenized using a heuristic component that tries to match the tokenization used in treebank data (for English, the Penn Treebank). Name finding, parsing, nominal classification, and proposition finding are described in the following sections. The value finder identifies ACE value strings like dates, times, and monetary amounts. Sentence-level relation-finding involves a preliminary run of the relation model for pronoun typing. For example, while “their” in “their goals” could be of type Person, Organization, or GPE, in “their sister”, the type would have to be Person. When this information is available, it can reduce the pronoun co-reference model’s search space.

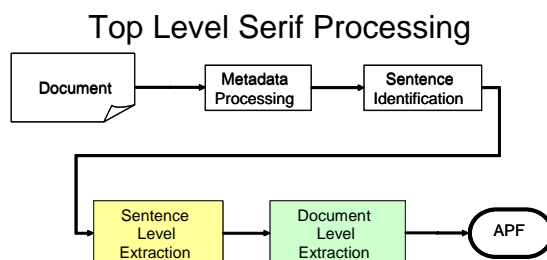


Figure 4.6: Top Level SERIF Dataflow

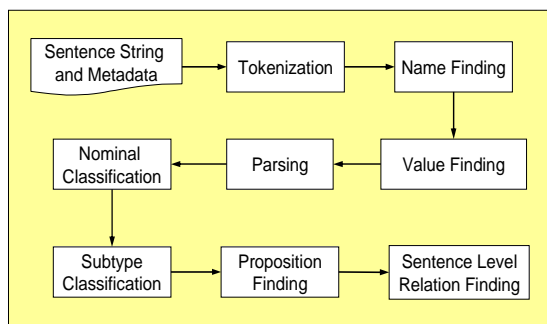


Figure 4.7: Sentence Level SERIF Processing

SERIF supports beam search to track multiple interpretations for sentence level processing, thus reducing the chances that a suboptimal choice by an early stage model will prevent the system from finding the best overall interpretation. This capability has been used to improve the integration between name finding and tokenization for Chinese, and also to prefer alternative parses that result in more ACE relations. The beam search capability is limited to the sentence level by design, since it is rare for “garden path” alternative interpretations to remain ambiguous beyond sentence boundaries.

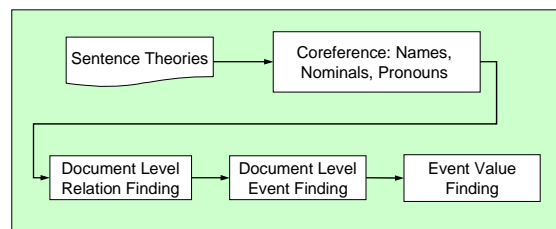


Figure 4.8: Document Level SERIF Processing

Once all sentences have been processed, SERIF executes the document level processing shown in Figure 4.8. The co-reference, event, and relation models are described below.

4.4.1.3.1. SERIF Model Types and Features

Though SERIF's parser uses a generative model, most of the other components use discriminative models, either maximum entropy models or log-linear ones trained using averaged perceptron training (Collins 2002). These discriminative models use large numbers of binary features to model context. Some (e.g., the relation and co-reference models) apply to a single instance. Others (like the name model) are sequence models that predict an interpretation for an interdependent sequence of instances, selecting the most likely path over a lattice of possibilities.

Many models use word-based features, and use a word clustering approach that extends the lexical coverage of such features using a word similarity measure, automatically derived from a large, unannotated corpus. This approach follows Brown *et al.* (1992), where words are clustered based on their distributional similarity. Clustering begins conceptually with each word in its own class, and then looks for pairs of words that can be combined into the same class with the smallest loss in average mutual information. BBN uses this approach to organize the entire vocabulary as a binary cluster tree. For example, "CPT" (an Irish corporate designator like "Inc.") may not occur in the name training, but may still occur in the large clustering corpus, and should cluster near other corporate designator words that occurred in the name training. A word cluster feature would then fire when a word in that region of the tree was assigned an ORG tag. During training, that feature would receive weight from corporate designator words with that tag, and then at run time, it could help predict that "CPT" should be tagged as part of an organization name.

Note that cluster size will be important in determining the usefulness of a feature. A cluster that is too small may not include enough terms that occur in training to receive much weight. A cluster that is too large will tend to include quite diverse terms, and the resulting positive and negative examples during training will tend to cancel out, leaving it with relatively little weight. To allow the model to automatically learn which size clusters work best, feature templates are created for various sizes, expressed in terms of levels in the cluster tree.

BBN has reported experiments (Miller *et al.* 2004) showing that word cluster features like these when combined with active learning techniques can allow name models with equivalent performance to be trained from only a quarter as much name-annotated

training data as would normally be required. SERIF makes extensive use of such word cluster features not only in its name models, but also in its models for nominal classification, co-reference, relations, and events.

4.4.1.3.2. Preprocessing

English tokenization is fairly straightforward, and is implemented using handwritten heuristics. Trained tokenization models for Chinese and Arabic assign tags to mark which characters start new words versus continuing previous ones.

4.4.1.3.3. Name Finding

SERIF uses a perceptron-trained log-linear sequence tagging model to recognize names. The tag set for the model includes X-ST (for “start”) and X-CO (for “continue”) tags for each name class X, along with NONE-ST and NONE-CO tags for words that are not part of any name. This approach distinguishes between two contiguous names and a single longer name of the same type, but does not handle nested names. In ACE, names are non-decomposable. Miller *et al.* (2004) describes this component in further detail.

4.4.1.3.4. Parsing

The BBN Parser implements a generative lexicalized PCFG model (Miller *et al.* 2000) similar to other current statistical parsers (Charniak 2000; Collins 2003). Trained on parse trees from the Penn Treebank (Marcus *et al.* 1993), the model estimates the probability of a particular parse tree as the product of the probability of the modifier attachment steps required to assemble the tree.

The generation of the right-hand-side (RHS) of a rule is broken up into a series of smaller steps, with a specified “head” child generated first, followed by its premodifiers and postmodifiers. The head word of the head child of a constituent is also carried along with each node as its lexical head, and possible attachments are evaluated based in part on this lexical information.

The generative process for each constituent begins by choosing its syntactic type, followed by choosing the POS tag of the head word and then the actual head word with its features (unless this constituent inherits its head word from its parent). The head child is then generated, followed by its premodifiers and postmodifiers.

The probability at each step is computed using maximum likelihood from the counts of similar events seen in training. Thus for example the constituent type of a modifier is conditioned on the constituent types of the parent, of the head constituent, and of the previous modifier in the given direction, as well as, on the head word of the parent. In “ate the apple”, the probability of an NP as a postmodifier of the VP with head “ate” would be based on the tuple VP (for the type of the parent), VBD (for the POS tag of the head), START (signaling that there is no previous postmodifier), and “ate” for the head word of the parent.

To handle data sparseness, the prediction based on all features is combined with backoff model predictions using feature subsets, e.g., the set of all features minus the head word feature. In some cases, multiple backoff stages are used with mixture weights computed using a modified version of the Witten-Bell formula.

The search for the most likely parse is organized bottom up, using beam search over a CKY-style chart. To ensure that the parse will be consistent with earlier results from the name model, name spans are initialized in the chart as pre-parsed leaves. (More data is typically available to train the name model, because name annotation is less expensive than Treebank style annotation.)

The parser can return an N -best list of alternative parses, rather than just a single best parse, with heuristic tests used to prefer alternatives that differ in ways likely to be significant for extraction.

For languages or domains where Treebank training data is not available, SERIF also supports an alternative partial parsing mode where a noun phrase chunk model is substituted for full parsing.

4.4.1.3.5. Propositions

SERIF extracts propositions from the parses to represent their semantic import (including verb/noun argument structure and modification relations). Later models like the relation model use these propositions as predictive features, without enumerating the different ways in which those propositional connections can be encoded in the underlying parse tree.

The proposition finders for each language are currently heuristic components that output a set of relations that use numbered entity arguments (e1, e2, ...) combined with lexical items taken from the text. For example, the propositional representation “Sam likes fudge” is shown in Table 4.4.

p1 = likes<verb>(<sub>:e1, obj>:e2)
p2 = <name>(<ref>:e1, "sam")
p3 = fudge<noun>(<ref>:e2)

Table 4.4: Propositions for 'Sam likes fudge'

The first proposition identifies the subject and object entities for the verb “likes”; the “<sub>” and “<obj>” labels are logical, even if the surface grammatical form is passive. The second proposition relates entity e1 to the string “sam” that is its name; the “<ref>” argument label identifies the relevant entity mention. The third proposition asserts the property “fudge” of the object entity e2, which comes from the head word of its noun phrase mention.

p35 = <name>(<ref>:e96, "yury zinin")
p40 = shot<verb>(<obj>:e96, in:e97, while:p39)
p36 = back<noun>(<ref>:e97)
p39 = leaving<verb>(<sub>:e96, <obj>:e98)
p38 = block<noun>(<ref>:e98, <poss>:e96)

Table 4.5: Propositions for 'Yury Zinin was shot in the back while leaving his apartment block'

There are copula, modifier, possessive, set, and comparative predicate types. Arguments typically represent entity mentions, but can also represent other propositions

(using their label, like “p1”) or literal text (as in the object of “say” in “Mary said no”). The parse for the sentence “Yury Zinin was shot in the back while leaving his apartment block” results in the propositions shown in Table 4.5.

4.4.1.3.6. Nominal Mentions

After predicting names, parses, and propositions, SERIF processes mentions to identify and type nominals and pronouns. The first step identifies “structured mentions,” those that combine primitive mentions in ways that call for special processing such as appositives, lists, or partitives.

A discriminative classifier predicts either one of the target semantic types or NONE for non-structured mentions. The model uses contextual features including word clusters, WordNet hypernyms (for generalization), and features based on word endings. Other features based on propositions can help predict, the semantic type of subjects, for example the subject of “say”.

4.4.1.3.7. Co- Reference

The co-reference process, which links together those mentions within a document that refer to the same entity, is organized into three stages covering name, nominal, and pronominal mentions.

The name and nominal co-reference models are both discriminative, feature-based models that operate left-to-right through the text, deciding for each instance whether to treat it as the first mention of a new entity or to connect it to one of the entities previously mentioned.

The pronoun co-reference model makes a similar sweep through the text, but also allows for linking a pronoun forward to a name or nominal mention later in the same sentence. Discriminative and generative models are available (Ramshaw *et al.* 2001), with features that include a distance measure that counts constituents in the order suggested by Hobbs (1977), along with a linear distance feature and features based on words and on the predicted mention types, number, and gender.

4.4.1.3.8. Relations

Relations link two entities. SERIF’s primary relation model is a classification model that is applied to each pair of mentions in a sentence to predict either a particular relation type or NONE as most likely, based on a set of contextual features and their trained weights. Relation model performance has also been improved by incorporating a number of subsidiary models, either as separate models whose output is then used as an additional feature for the main model, or by heuristically combining the predictions of the two models to produce a more optimal joint prediction.

4.4.1.3.9. Events

Events in SERIF can have arbitrary numbers of arguments, including both ACE entities (e.g., people) and non-entity values (e.g., dates). For each event, a single trigger word or phrase is annotated. SERIF’s event model works in three stages: (1) identify

possible event triggers, (2) attach likely arguments to each possible trigger, and (3) decide which trigger + argument clusters to report, merging those that refer to the same real world event.

The event trigger stage intentionally favors recall over precision, e.g., tagging “hit” as an attack trigger in both “bombs hit Iraq” and , “hit a home run”. Triggers are pruned in the third stage, after potential arguments have been assigned.

The primary trigger-finding model is a perceptron-trained discriminative classifier. The model uses features about the word and its context, including proposition features, word cluster and WordNet synset features, and document topic features. Other trigger models use simple maximum likelihood estimation to identify cases where the addition of the subject or object argument strongly increases the model’s confidence compared to a backoff model based only on the trigger word.

The second stage in the event model uses a discriminative maximum entropy model to determine the probability of each entity or value in a certain role (e.g., Agent, Artifact, or Time-Within) in a possible event. The model’s features capture both the surface-level and deeper syntactic connections between the trigger and potential arguments.

The decisions about a trigger’s arguments are not independent. The resulting probability scores are used to select argument assignments for each trigger that maximize the total probability of the argument set while filling each role if plausible and restricting each mention or value to filling at most one role. For example, in the sentence “The defendant *moved* the guns”, “defendant” could be either the Agent or the Artifact of the move event. The presence of “guns,” as a likely filler for Artifact, will cause “defendant” to be classified as the Agent.

In the final stage of the event model, the candidate clusters of trigger words and arguments are filtered with a set of heuristics and combined to produce a final set of events. A maximum entropy model predicts the likelihood that, for each pair of trigger word clusters, the two represent the same event, using primarily match/mismatch features between the event arguments or trigger words.

Since these merging predictions are made independently, they can be inconsistent. For example, the model might predict that “the attack” should link to “the attack in the Spanish capital” link, and that the latter should link to “the incident in Madrid”, but still predict that “the attack” and “the incident in Madrid” should not link. In such cases, the link predictions are preferred over the no-link ones, because link decisions tend to be better informed than those not to link.

4.4.1.3.10. Cross-Document Co-Reference

BBN’s cross-document co-reference component links named entities across a corpus of documents. We divide the problem of cross document co-reference into two sub-problems *name variation* and *entity disambiguation* (Baron and Freedman 2008). The name variation component identifies name strings that could refer to the same entity. It uses measures of string similarity, lists of known aliases, and features pulled from SERIF’s output to produce lists of equivalent name strings. For each set of ‘equivalent names’, the entity disambiguation component runs an agglomerative clustering algorithm over all document entities that include one of the name strings in the set. Features used

during agglomerative clustering include predicted document topics, relations and events extracted by SERIF, and document metadata.

4.4.1.4. Evaluation Results

SERIF has been a consistent top performing system in ACE evaluations of entities, relations, and events in English, Chinese, and Arabic over the past five years. Complete results for ACE in 2007 and 2008 and are available on NIST's website (<http://www.itl.nist.gov/iad/mig/tests/ace/>). Table 4.6 provides ACE results for the highest, lowest, and average system in the entity detection and linking (EDR) and relation detection (RDR) tasks. These numbers reflect each site's primary submissions. The ACE scorer maps system-produced entities to reference entities. ACE value has been the official metric of the ACE evaluation. It puts a higher cost on certain entity classes (e.g., people are more important than facilities), certain mention classes (e.g., names are more important than pronouns). It penalizes systems for mistakes in linking, as well as, type and subtype detection. Incorrectly assigning a mention to an entity is very costly in terms of value score: for names, a system will be penalized 1.0 for the missed mention and an additional 0.75 for a mention false alarm.

	English		Arabic	
	<i>EDR</i>	<i>RDR</i>	<i>EDR</i>	<i>RDR</i> ⁴
Top	52.6	11.6	43.6	5.5
Average	-53.3	7.7	17.3	5.5
Low	-269.1	3.8	-9.1	5.5
SERIF	52.6	11.6	43.6	5.5

Table 4.6: Value Score for ACE 2008: LEDR and LRDR

4.4.1.5. Conclusions

SERIF's combination of trained models working over full syntactic and propositional analyses has produced state-of-the-art performance on the core multilingual information extraction task and has also provided a foundation for a wide range of related language understanding applications, from speech and OCR input through to query analysis and response.

SERIF's predictions of ACE-style extraction elements and core syntactic structure are critical to BBN's approach to Distillation. The extraction elements provide targeted document retrieval for specific entities and certain classes of events. We identify relevant answers using a rich pattern language that combines the extraction elements with propositions (see Section 4.3.2.6). The propositions are also used in a distance metric that identifies spans of text that are similar to a topic or event specified in the query (see Section 4.4.4.4). Extending these results going forward will call for richer, more integrated models, but building on the same foundation.

⁴ BBN was the only participant in Arabic RDR.

4.4.2. Statistical Sentence Extraction for Multilingual Information Distillation

Authors: Dilek Hakkani-Tür, Gokhan Tür, Michael Levit, Daniel Gillick, Adish Singla, and Sibel Yaman

4.4.2.1. Introduction

Our Distillation approach is based on using document retrieval for finding relevant documents in response to a given query, and statistical classification to extract sentences as Snippets from the relevant documents (Hakkani-Tür and Tur 2007). The set of selected sentences is then reduced by removing redundancies. Here, we focus on sentence extraction for information distillation using binary classification. To train sentence extraction models, negative and positive (relevant) examples are computed from manually annotated answer keys, which have the relevant snippets and the corresponding document identifiers for each query. Then, possible variants of all names in each sentence and query are found and included as alternative spellings. Furthermore, a set of syntactic and semantic features is computed from these sentences, and a graph structure, that is used for feature generation, is formed (Levit *et al.* 2007). During sentence extraction from non-English documents, in addition to automatic translations of the candidate sentences, word sequences from the original documents are used and have been shown to be beneficial experimentally (Singla and Hakkani-Tür 2008). Section 4.4.2.2.5 presents methods for combining these information sources during classification.

4.4.2.2. Sentence Extraction

The first step in sentence extraction is retrieving documents that are relevant to the given query. We use the University of Massachusetts Indri search engine (Strohman *et al.* 2005) to find the relevant documents. Then the documents are segmented into sentences by the NYU ACE System (Grishman *et al.* 2005) for text and the ICSI+ sentence segmentation system for speech input (Zimmerman *et al.* 2006). While a similar information retrieval based approach may be used for the sentence extraction, we employ discriminative classification methods, for robustness to the noise introduced by the automatic speech recognition and machine translation systems for non-English and/or audio documents.

4.4.2.2.1. Classification-Based Approach

To train sentence extraction models, we extract negative and positive examples from the given answer keys, which have the relevant snippets and the corresponding document identifiers for each query. All sentences in the relevant documents are used as examples; and the sentences that are fully or partially in the answer keys are marked as positive examples and all the rest as negative examples. In this step, the same sentence can be marked as a positive example by one query and as a negative example by another. For building models from audio sources, we use speech recognizer output and align the automatic hypotheses with reference sentences to extract their class (positive or negative)

from the answer keys. Given the collection of all relevant sentences, x_1, \dots, x_m , and irrelevant sentences, y_1, \dots, y_n from each document $d_j \in D$, where D is the set of documents that have at least one snippet as an answer to a query, we form the set: $S = \{(x_1, 1), \dots, (x_m, 1), (y_1, 0), \dots, (y_n, 0)\}$

Then the sentence extraction for the distillation task is defined as estimating the conditional probability, $P(c|s_i)$, $c \in \{0, 1\}$, that sentence s_i is relevant ($c = 1$) or not ($c = 0$). The sentences that have $P(c|s_i) > \theta_q$, where the threshold θ_q is estimated using a held out set for each query q , are then output by sentence extraction.

4.4.2.2.2. Finding Variants of Names

Proper names, especially names of foreign origin, are often written in different ways. For example, Wikipedia indicates that over 30 different spellings for the name of the Libyan leader, *Muammar al Gaddafi*, have appeared in published media.⁵ This variability creates a word matching problem for language processing and a data fragmentation problem for statistical language modeling used in speech recognition and machine translation, because variance of the same name may not be recognized as such. Since our sentence extraction approach relies on matching words, we automatically detect variants of names that refer to the same person using a novel unsupervised method for learning spelling edit probabilities. This method results in overall F-measure improvements on name variant detection experiments we performed on an in-house data set (and achieved around 0.80 F-measure), as well as, improvements on sentence extraction for distillation (Gillick *et al.* 2008).

While finding variants of names is often addressed when forming databases, where entities are stored in structured fields (Cohen *et al.* 2003; McCallum *et al.* 2005), we consider its application to unstructured text. The name variant detection algorithm draws inspiration from statistical edit-based spell checking, developed by Kernighan *et al.* (1990) and improved by Brill and Moore (2000). To improve on standard edit-distance, we introduce edit probabilities, which include character and word substitutions, and abbreviations. We use the following unsupervised bootstrap method for learning the edit model: We find all pairs of names that differ by exactly one edit and assume that their match probability is computed using a TF.IDF based word context window similarity. Then the probability of each edit can be estimated by maximum likelihood. Using the resulting probabilistic edit model improves F-measure significantly. This algorithm and related experiments are described in detail in Gillick *et al.* (2008).

4.4.2.2.3. Syntactic and Semantic Graphs

In our approach, each sentence is represented as a directed acyclic graph, called a *Syntactic and Semantic Graph* (SSG) or *Chart*, which is an amalgam of elements from various annotation levels (Hakkani-Tür *et al.* 2005). An SSG of a sentence with N words consists of $N+1$ states that are connected by arcs labeled with SSG entries. Graph entries can be of several types. The simplest ones are words, so the trivial graph is just a single

⁵ <http://en.wikipedia.org>

path whose arcs are words (in their natural order). Other annotation levels can be integrated in SSGs as well. In this work, we consider the five SSG entry types:

- **Words**
- **Part-of-speech tags and syntactic parses**
- Arguments and targets of PROPBANK **predicate-argument structures** (Kingsbury *et al.* 2002)
- **Information extraction elements** including entities, relation and event arguments with co-references
- **Slot** values, which are those parts of each sentence which correspond to the variable portion of a given query.

For example, for the query in Figure 4.9, the slot value for the variable portion of the query is “Baghdad”. Therefore, the word Baghdad is marked with a SLOT arc in the SSG of each sentence considered as a candidate answer for this query.

These annotations are inserted into SSG as arcs parallel to the words they correspond to and span the same nodes with them. The SSG for sentence “*John gave Mary his car*” is shown in Figure 4.10, where arc prefixes indicate the SSG entry type. More information on the computation of SSGs is presented by Levit *et al.* (2007).

Query: Describe attacks in [Baghdad] giving location, date, and number of dead and injured. Provide information since [28 Sept 2008].

Snippets:

- *Attack against a school bus filled with children*
- *There were 45 students and 2 teachers in the bus*
- *No organization claimed responsibility*

Figure 4.9: An example query and answering snippets.

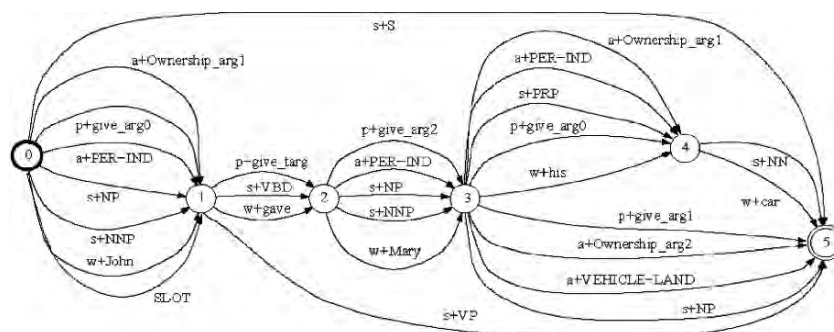


Figure 4.10: Syntactic and semantic graph for sentence “*John gave Mary his car*”; it contains 5 words (prefix w+), 10 syntactic nodes (prefix s+), 4 mentions of 3 entities (prefix a+), arguments of one *ownership* relation, and finally target and arguments of verb *give* (prefix p+)

While most of the SSGs are computed offline, they are augmented with slot features once the query is received. To find instantiations of query slots in a sentence, we use weighted bag-of-words pattern matching, with the name variants described earlier, some

linguistic knowledge (like verb nominalizations and synonyms), and world knowledge (such as gazetteer resources). Contributions of individual words to the overall score depend on their parts of speech and occurrence frequencies. For example if the location is “Iraq”, the word “Baghdad” is labeled also as “slot” with some weight. Similarly, if the query is about “George Bush”, but we have “Bush” mentioned in the sentence, it gets a much higher slot weight than “George” mentioned in the sentence. Furthermore, if the slot is “Qaddafi” but the sentence has a variant of it, say “Kaddafi”, a weighted slot arc is also introduced. These weights are optimized on a held out test set for each template.

4.4.2.2.4. Feature Generation

Two types of features are extracted from SSGs: *n*-grams and inclusion features.

n-gram Features

Since SSG arc labels of all types are treated the same way, *n*-grams extracted from them can consist of entries that are quite heterogeneous in nature. For instance, from the SSG in Figure 4.10, the following *n*-grams can be extracted (we use “ \cup ” as a connector in *n*-gram representations):

$$\begin{aligned} a+PER \cup p+give \text{ targ} \cup w+Mary \\ w+gave \cup a+PER \cup s+NP \\ w+John \cup s+VP \\ a+PER \cup s+VP \end{aligned}$$

For example, an *n*-gram such as “bombed SLOT” would be a strong indicator of an attack in the requested location entered as the query slot value. This generation process is very powerful; paired with a task-dependent feature selection process, it allows keeping only those features that strike just the right balance between specificity and generality, while ignoring everything else.

For instance, if the question is about John’s actions, the third feature in the list above seems most relevant. If we are asked about people’s actions in general, the fourth feature will probably suit our needs better. Finally, if the focus is on the process of giving, the first feature in the list should be favored with respect to the others.

Inclusion Features

While *n*-grams contain information about sequences of chart entries, the inclusion features focus on parallelism. For example, the word *John* is at the same time a noun phrase and an entity of type PERIND (person, individual), but also the first argument in a relation and arg0 (agent) for the “giving” predicate. These features are obtained by computing all pairs of chart entries that “contain” each other (in terms of the span of the words they account for). In the chart in Figure 4.10, some of the inclusion features will be (“ \ni ” stands for “includes”)

$$\begin{aligned} a+Ownership \text{ arg1} \ni a+PER-IND \\ s+S \ni s+VP \\ a+VEHICLE-LAND \ni w+his \end{aligned}$$

SLOT values are also helpful with inclusion features. For example, knowing that the name mentioned in the person name slot is the object (arg1) of the predicate “arrest” would help a template questioning arrested people.

4.4.2.2.5. Cross-Lingual Processing

In cross-lingual distillation, the user query is in one language (usually called *target language*), and documents are in other languages (called *source language*). In such a case, one can either search the query in source language (in this case, Arabic or Mandarin) documents and then translate the output to target language (in this case, English) or translate the source language documents to target language and search the machine translation (MT) output, or search both and combine the outputs.

Since the presented approach is largely language independent, we also explored techniques for cross-lingual sentence extraction for information distillation from non-English sources and combined the two systems working on the source side and target (English) side. Usually, there are more natural language processing (NLP) resources and tools on the English language side. However, the noise introduced by MT into the documents usually degrades the performance of following NLP tasks. There are at least three ways of using information from both sides, as shown in Figure 4.11:

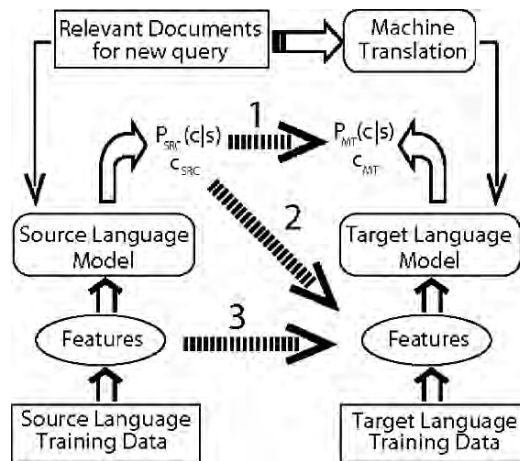


Figure 4.11: Three ways of combining sentence extraction in source and target languages

1. **Posterior Probability Interpolation:** For each class $c \in \{0, 1\}$ and sentence s , the posterior probability estimated by the two models, $P_{SRC}(c|s)$ from the classification model trained from the source language data (that is, *the source language model*) and $P_{MT}(c|s)$ from the classification model trained from the target language data (that is, *the target language model*) can be interpolated, estimating the new probability:

$$P(c|s) = \lambda \times P_{MT}(c|s) + (1 - \lambda) \times P_{SRC}(c|s)$$

where $\lambda \in [0, 1]$ is estimated from a held-out data set.

2. **Using the output from one model as a feature in the other:** The probability estimate, $P_{SRC}(c|s)$, and/or the class estimate \hat{c}_{SRC} from the source language model can be used as a feature in the target language model (or vice versa).
3. **Combining features from Source and Target Language Sentences and MT output:** Instead of training two separate models from source and target language data, the set of features computed from the two data sets can be combined and a single model is learned from them. Assume S_i is the set of features extracted from the sentences in the source language, T_i is the set of features extracted from the machine translation output corresponding to S_i , and c_i is the relevance annotation (where $i = 1, \dots, N$, and N is the number of examples from source language documents). Then one can learn a model from $(S_i \cup T_i, c_i)$, instead of the two models learned from (S_i, c_i) and (T_i, c_i) . To utilize the answer keys from English documents, with features E_j (where $j = 1, \dots, M$, and M is the number of examples from English documents) and relevancy annotations e_i , one can treat English examples in the same framework by considering unknown values for source language features. The classification algorithm used should be able to deal with unknown feature values for this case.

These approaches are described in detail by Singla and Hakkani-Tür (2008).

4.4.2.3. Experiments and Results

#	Text
1	List facts about event: [EVENT]
8	Describe the prosecution of [PERSON] for [CRIME]
12	Provide a biography of [PERSON]
15	Identify person arrested from [ORGANIZATION] and give their name and role in the organization and time and location of arrest
16	Describe attacks in [LOCATION] giving location, date, and number of dead and injured

Table 4.7: Query templates considered for system evaluation

We focus on sentence extraction for five types of queries used for the DARPA GALE Program, listed in Table 4.7.

4.4.2.3.1. Experiments with English Data

Documents that supplied answer keys for training and test queries came from different corpora distributed by Linguistic Data Consortium. We used the English text

part of the GALE training data (LDC catalog number: LDC2006E15) for training and the English text part of the GALE corpus (LDC catalog number: LDC2007E13) for testing. Overall, there was an average of 24 training and 9 test queries per template. About 2700 (500 for test) documents that were considered contained 38,000 (16,000 for test) sentences, and supplied an average of 72 (39 for test) relevant snippets per query. The test corpus had a significantly smaller proportion of relevant snippets (11% as opposed to 21% in the training corpus), and its queries were judged as subjectively more difficult by human observers. All documents were processed in advance to obtain syntactic parses and PROPBANK annotations using Charniak and ASSERT parsers (Pradhan *et al.* 2004). We used NYU’s toolkit JET (Grishman *et al.* 2005) to produce ACE annotations. Similar processing was applied to all queries, and in particular to their slot spans. Note that, in these experiments, we present results using the relevant documents in the answer keys, and not the ones returned by INDRI, as we do not have manual relevancy annotations for those.

To investigate the potential of using other annotation layers in addition to words, we created parallel sets of classification features extracted from SSGs that contained only words, as well as, features from SSGs with all annotation layers introduced above. Table 4.8 shows the F-measures for class “relevant” achieved for each template with a support vector machine classifier (Joachims 1999). The baseline results correspond to performance obtained by marking all candidate sentences as relevant.

template	1	8	12	15	16	Avg.
Baseline	0.40	0.32	0.17	0.22	0.33	0.29
words	0.42	0.42	0.25	0.40	0.46	0.39
all feats	0.47	0.51	0.43	0.42	0.52	0.47

Table 4.8: Classification results (F-measure) for a baseline classifier that marks all sentences as relevant (baseline), uses word bigrams, and all features (all feats).

We see that for all templates, introducing additional annotation levels to produce classification features resulted in very significant improvements. In these experiments we used bigrams, but using trigrams did not improve the performance. It is worth mentioning however that, given the words only baseline’s ignorance of the slots, its performance looks remarkably good. What happens is that the system assumes that the selected documents *are* about the slots, so that it needs to be looking only for words that indicate the right template. For instance, if an arrest is mentioned in the document, it is presumably an arrest of the slot-person. Another reason is most clearly articulated by the open-domain template 1. Here, the seemingly high F-measure for the word-only case is not much better than the one of a trivial classifier that marks all sentences as relevant (last row of Table 4.8) In fact, unlike other templates, entire selected documents are often dedicated to the event in question, so that most of the sentences in them are on-topic (e.g., for event “*Civil unrest in France*” this proportion is about 65%).

4.4.2.3.2. Cross-Lingual Sentence Extraction Experiments

To see the effectiveness of the cross lingual sentence extraction, we performed experiments using Template 16 from the GALE project Year 1 and Year 2 distillation data sets, since annotated data from both of the non-English data sources for this template is available. For the GALE Year 1 data set, there is a total of 24 queries for template 16; all have answer keys from English documents, 11 of them have answer keys from Arabic documents and 12 of them have answer keys from Mandarin documents. For the GALE Year 2 data set, there are 9 template 16 queries with answer keys for English, 8 for Arabic and 9 for Mandarin. The Arabic and Mandarin documents were translated into English using SRI’s machine translation system (Zheng *et al.* 2008).

We performed n -fold cross-validation with a total of 19 rounds for Mandarin data and 21 rounds for Arabic data, and used a boosting classifier (Schapire and Singer 1999); 20% of the training data is used as a validation set while computing the optimal number of iterations and probability threshold values, while data from one query is the test set for each round. To compare different systems, we use macro-averaged F-measure over query template examples. In this study, we used word (for Arabic and English) and character (for Mandarin) n -grams and no syntactic or semantic features.

Table 4.9 has the results from the experiments, where source and target language information is used. In the first combination experiment, the probabilities from the source and target language models are interpolated. The second and third combination experiments include the output from the source language experiment as a feature in the target language experiment. One uses the probability output from the source language classifier, P_{SRC} ; the other uses the relevancy decision, \hat{c}_{SRC} . The fourth combination experiment uses only data from the source language, but also includes features from MT output sentences. The last experiment is similar to the fourth, but the training data set is much larger, since the data from English sources (MT+ENG) is also used.

Method	Arabic	Mandarin
baseline	0.42	0.47
mono. (SRC/SRC)	0.60	0.60
mono. (MT/MT)	0.58	0.64
comb. 1	0.68	0.70
comb. 2 with P_{SRC}	0.58	0.67
comb. 2 with \hat{c}_{SRC}	0.62	0.63
comb. 3 (MT & SRC)	0.56	0.62
comb. 3 (MT+ENG & SRC)	0.58	0.65

Table 4.9: Cross-lingual distillation F-measure results, when two information sources are combined. MT represents the automatic translations of non-English data (SRC), ENG represents data from English documents. “mono.” Is used to denote mono-lingual experiments, that is, either source or target language examples are used in these.

As seen in Table 4.9 for both languages, one or more of the combination methods helped improve performance. The best results are obtained using linear interpolation for

both. We experimentally show that the combination approach results in about 9.3% and 13.3% relative F-measure improvement for Mandarin and Arabic respectively over the best monolingual baseline. The first combination approach of building two separate models and interpolating their probability outputs has given the best results for both Arabic and Mandarin.

4.4.2.4. Conclusions

This section described the ICSI-SRI approach that selects document sentences with respect to their relevance to a query using statistical classification methods. The contributions of our approach include, 1) a novel method to generate classification features extracted from syntactic and semantic graphs, formed using elements from various annotation layers, such as word transcriptions, syntactic and semantic parses, and information extraction annotations, 2) a probabilistic method to compute name variants, 3) and the use of source and target language data to train extraction models.

Acknowledgements

We thank Jing Zheng for providing us with the machine translation output used in the experiments and the SRI GALE Distillation team members at Columbia University, University of Massachusetts, NYU, and Fair Isaac.

4.4.3. Snippets: Using Heuristics to Bootstrap a Machine Learning Approach

Authors: Daniel M. Bikel, Vittorio Castelli, Radu Florian, Xiaoqiang Luo, Scott McCarley, Todd Ward, and Imed Zitouni

4.4.3.1. Introduction

We address the problem of identifying collections of snippets (i.e., sentences, or selected portions thereof) that are relevant to the answer to a structured GALE query. We propose an approach to this task that consists of seeding a statistical rule-based system. Specifically, the rule-based system is used to bootstrap the annotation of training data for the statistical system.

Our approach has two main motivations. First, generating a heuristic system is an efficient way of building a working prototype that can guide the development of a statistical system with better F-measure characteristics. Second, the heuristic system can be used to filter large amount of data and produce a restricted corpus for annotation that contains a much higher percentage of positive examples than the full data set.

To evaluate the approach, we compare the statistical system to a finely-tuned rule-based system on the snippet selection task of our GALE system. This task is at the heart of the GALE program distillation. Its purpose is to eliminate all the sentences that are not relevant to a given query and to assign a relevance score to the remaining ones. In a full-blown GALE system, the scored snippets are then collectively analyzed to identify and

remove content redundancy. The final output of the system consists of the most relevant subset of snippets that contain no redundant information.

Rule-based methods have long been used in natural language processing. They have low start-up costs, are easy to implement — by an experienced architect — with a relatively small number of examples of the task, and yield good results on structured or semi-structured data. They are computationally efficient, and, to some extent, the user can visually inspect traces of their execution and explanations of how the results were selected. On the down side, rules are difficult to write if the data are not structured, or are even moderately noisy (e.g., or automatic transcripts). Rule-based systems additionally suffer from scalability issues. The primary cause is the difficulty in discovering the positive and negative interaction between rules, and building the control structure to appropriately leverage them. Finally, it is difficult to write effective rules that fire based on real-valued numbers, and in particular probabilities, within a rule-based system. This limitation makes the integration of probabilistic information sources (such as POS taggers and mention detectors) particularly challenging, especially when multiple such sources are combined by individual rules or chains of rules. For the past 20 years, statistical modeling has become the dominant approach to natural language processing (Church 1988). The advantages of statistical methods include their ability to combine arbitrary types of input (including the output of rules) and their very good empirical performance. Statistical systems can be easily retrained when new data becomes available. On the down side, they depend on the existence of medium to large annotated corpora, which require a precise definition of the task at hand and substantial amount of human annotations effort.⁶ Historically, statistical systems have been criticized as being difficult to inspect; however, these criticisms have eased over time as new methodologies for inspection have emerged in the literature (Lee 1999; Bikel 2004).

Hybrid approaches have also been proposed, notably transformation-based learning (Brill 1992), which attempts to combine the simplicity and inspectability of rule-based approaches with corpus statistics and has been successfully applied to selected tagging problems.

4.4.3.2. Task Description and System Architecture

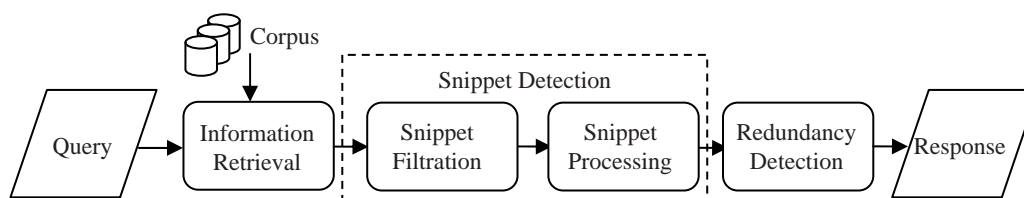


Figure 4.12: Rosetta Distillation System Diagram

The snippet relevance detection module, which embodies the snippet-selection methods described in this section, is part of Rosetta’s distillation pipeline, presented in

⁶ It is important to mention, however, that the annotation does not require experts, but rather good speakers of the language.

Figure 4.12. We preprocess the documents in the distillation corpus with our information extraction system, BlueMax (Florian *et al.* 2004). BlueMax performs sentence segmentation, tokenization, and parsing; identifies numerous entity mentions, including PERSON, AGE, and OCCUPATION; and performs co-reference resolution and relation extraction on these mentions. Additionally a cross-document co-reference resolution component assigns global IDs to PERSON and COUNTRY entities.

After preprocessing the corpus, we use a five stage pipeline, consisting of query preprocessing (including processing the arguments of each query using BlueMax), document extraction using CMU extensions to the INDRI search engine, snippet filtration, snippet processing and planning (redundancy detection). Snippet filtration and snippet processing are the two components of the *Snippet Detection* box of Figure 4.12, the main focus of this section. Their purpose is to select snippets that are relevant to the current query.

Determining relevance precisely is a difficult exercise in natural language understanding. Subjectivity makes it challenging even for humans to perform this task consistently. This is exemplified by a simple experiment based on template 17, where we recorded an inter-annotator agreement F-measure of 0.82.

Our approach estimates relevance by computing a score that captures the following situations:

- 1 The snippet contains a direct or indirect mention of the query argument(s). For queries about a PERSON or an ORGANIZATION, we test if the snippet contains the PERSON or ORGANIZATION name, or a nominal or pronominal mention of the name. For queries about a TOPIC or an EVENT, we test if the snippet contains a description of the target topic or event.
- 2 The snippet contains query-specific information. For example, for the biography template, the snippet must contain the person's age, birth place, profession and/or education.

For example, if the query is “List attributions by Hu Jintao on George Bush”, a relevant answer should contain a quotation, the quotation must be attributed to Hu Jintao, and the topic of the quotation should involve George Bush. A joint statement by both presidents would not satisfy the query in spite of displaying several traits of a correct answer.

The Snippet Detection Systems described in the following sections address the challenge by relying on a large variety of lexical, syntactic, and semantic information, including:

- morphological information
- part-of-speech tags
- word senses (WordNet-based)
- mention detection information (32 types of mentions, including persons, organizations, locations, countries, dates, etc., are detected) and event information (13 types of events are detected)
- in-document and cross-document co-reference

- parse information and semantic role labels
- date resolution

4.4.3.3. Rule-Based System

A version of the Rosetta Distillation System relies on a rule-based snippet detection stage. In this section we describe in detail how rules were written and how empirical weights were assigned to the rules.

4.4.3.3.1. Rule Writing and Scoring

Recall that a snippet is considered relevant to a query if 1) it contains mentions of the arguments and 2) it contains query-specific information. Correspondingly, we rely on two types of rules: one for the query arguments, and one for the query-specific information. We assign to each rule r_i a positive deterministic score $s(r_i)$. Scores greater than one denote rules that fire in the presence of positive evidence, while rules with score less than one fire in the presence of irrelevant snippets.

For example, rule r_5 in Table 4.10, which detects the presence of an OCCUPATION mention in apposition to a mention of the target person, provides positive evidence for the biography template, and has a high weight. Conversely, rule r_7 , which detects the presence of a LOCATION mention near target person, is not as discriminative as r_5 for the same template, and has lower weight.

ID	Rule	Weights
r_1	contains the target name?	4.0
r_2	contains a nominal or pronominal mention of target name?	2.0
r_3	does not contain the target name?	10^{-10}
r_4	contains AGE in vicinity?	8.0
r_5	contains OCCUPATION in vicinity?	4.0
r_6	contains DATE in vicinity?	2.0
r_7	contains LOCATION in vicinity?	2.0

Table 4.10: Sample rules and associated weights

Let $R(p)$ be the set of rules that fire on a given snippet p . We then compute the score $s(p)$ of the snippet as:

$$s(p) = \prod_{r \in R(p)} s(r)$$

The candidate snippets $C(q) = \{p_i: i = 1, 2, \dots, N\}$ of a query q are then ranked based on the assigned scores: $(p_{(1)}) \geq S(p_{(2)}) \geq \dots S(p_{(N)})$, where the subscripts in parentheses denote rank order.

4.4.3.3.2. An Example

In this section, we use a biography query to illustrate how the rule-based snippet ranking system works:

PRODUCE A BIOGRAPHY OF [Canadian Prime Minister Stephen Harper].

Some sample rules are listed in Table 4.10. Rule r_1 , r_2 and r_3 are to test if a snippet contains the PERSON argument: Canadian Prime Minister Stephen Harper. Note that the Distillation System must be robust to errors made by the BlueMax information extraction system, and in particular to mistakes in entity detection and co-reference resolution. For example, we observed that matching a person by name is much more reliable than by nominal or pronominal mention in the co-reference chain. Thus we assign a higher weight to Rule r_1 than r_2 . Rule r_3 is an example of a negative rule: if the target person cannot be found either by lexical matching, or by co-reference chain, the snippet is assigned a very low weight, which effectively eliminates it from consideration.

The top three snippets for Query 3.2 are list below in descending order of relevance score. Note that some English documents are from machine translation, and contain ungrammatical sentences; the Distillation System is designed to retrieve answers from both “clean” text and MT output.

- 1 The ruling General requested Canada Michael Jean yesterday evening Tuesday of the *Governor* Stephen Harper who won in the elections yesterday, Monday, to form the new government.
- 2 Using ominous martial music and stern voiceovers, the governing Liberal party has released television commercials suggesting that Stephen Harper, the ***Conservative leader***, supports the Iraq war, conspires with Quebec separatists and may even receive illegal donations from American conservatives.
- 3 Unless every national poll here is amiss, Stephen Harper, *46, an economist and social conservative* who is writing a history of ice hockey, appears poised to lead his Conservative Party to victory over Prime Minister Paul Martin’s Liberal Party, something that seemed highly improbable just a few weeks ago.

The first and second snippets receive high scores because they both contain an OCCUPATION (Rule r_5) mention in the vicinity of the query argument. The first mention gets higher score than the second one because it also contains a DATE (Rule r_6) mention. The third snippet contains both AGE and OCCUPATION mention, and is thus assigned high score.

The following are examples of snippets that contain mentions of “Harper,” but are assigned low scores because they lack biographic information:

1. But Harper told them that often those aspirations are neither focused nor built on a firm foundation of preparation.
2. The White House was announced Friday on a visit Harper, but he did not specify the agenda topics.

3. Powell to visit him, Harper official however Hall this morning where asked him officially, the formation of a government.

4.4.3.3.3. Challenges

As shown in this section, a rule often tests if a snippet contains a very specific semantic item (e.g., occupation, age, etc.). Once a set of good rules are written, the system can produce relevant snippets with high precision. However, the rule-based approach suffers from several shortcomings:

- The rule scores are constructed empirically, via a process more akin to art than to science: there is no established process to construct good rules and assign proper scores other than trial-and-error and experience.
- Score assignment becomes increasingly difficult as the number of rules grows large.
- Scalability with respect to the number of supported templates is hampered by the dependence of rules and, especially, of weights on the specific template.
- Maintenance and upgrade cost grow dramatically with the number of rules.

4.4.3.4. Statistical System

In this section, we focus primarily on the snippet filtration and snippet processing phases.

4.4.3.4.1. System Architecture

The overall system architecture consists of a cascade of statistical models, organized hierarchically in a directed acyclic graph. The two main methods for constructing a cascade of statistical models are: 1) feeding the output of previous stages to the current stage, and 2) using the output of previous stages as features for the models of the current stage. We employ both types of cascading.

4.4.3.4.2. Snippet Processing

The input queries are template queries with one, two, or three instantiated arguments. Examples:

- Describe the relationship between [PERSON|ORG] to [PERSON|ORG].
- Find statements by [PERSON] on [TOPIC].
- List facts about [EVENT].

The problem of answering queries like these may be broken down into smaller sub-problems, the components for which may be efficiently reused. For example, several query types involve arguments of type PERSON; therefore, it would be very useful to have a binary classifier that determines whether a particular PERSON from a query argument appears in a snippet, either nominally or pronominally, and similarly for ORG's (organizations). An EVENT or TOPIC can be a noun phrase, verb phrase or entire

inflectional phrase that itself contains instances of PERSON's or ORG's. In such a case, not only would it be useful to have a binary classifier that detects whether a particular EVENT or TOPIC appears in a particular snippet, but such a classifier would surely benefit from the output of the PERSON or ORG classifiers if the event involves a person and/or an organization.

Accordingly, our system employs *argument models*, binary classifiers that detect whether query arguments appear in snippets, and *template models*, models that detect whether a snippet is a suitable answer to a query under the GALE distillation rules. Crucially, some argument models employ other argument models, and template models employ argument models, as well as, other template models.

It quickly became essential for us to experiment with a wide variety of features and feature combinations, which included using the output of one model as a feature to another. To achieve this goal, we employed a Java scripting language called BeanShell (Bea, 2009). Determining which features and feature conjunctions to use and how those feature functions would operate could be largely controlled by editing a BeanShell script. The system would read these BeanShell scripts at runtime, allowing us to explore a wide variety of models with great efficiency.

4.4.3.4.3. Corpus Annotation

In order to build our binary classifiers, we had to annotate data. After a pilot study to determine feasibility, we used our deterministic system to retrieve a limited number of snippets that were likely to contain positive results. We did this so as to avoid having too many negative examples to annotate. After the first round of annotation, we developed a set of features for the initial statistical classifier, and we used a subset of these features as filters to create the actual dataset: examples for which any of the selected features fired were retained for annotation. Table 4.11 shows that amount of data annotated for each argument type.

Model	Sentences	Tokens
PERSON	1698	46,481
ORG	11,765	226,574
LOCATION	4,912	119,987
COUNTRY	19,994	454,444
EVENT	4,747	170,944

Table 4.11: Amount of data annotated for each argument model

For the PERSON, ORGANIZATION and LOCATION models in particular, it was important to annotate instances that were frequent enough to have meaningful statistics, but not so frequent as to be trivial.⁷ Accordingly, we annotated PERSON instances whose names appear in exactly 10 documents in the entire corpus.

⁷ With trivially easy instances, such as “Hillary Clinton”, our model would only end up learning to trust the output of the BlueMax mention detector, which is encoded as a feature.

4.4.3.4.4. Algorithms

Our information extraction engine, BlueMax, uses maximum entropy (MaxEnt) modeling and Viterbi decoding for almost all sequence-based classification problems, such as tokenization, part-of-speech tagging/parsing and mention detection. We also use a MaxEnt model to do co-reference resolution (Luo *et al.* 2004).

Name	Description
matchName	True if the PERSON argument text appears exactly as-is in the snippet text.
matchXdcId	True if the cross-document co-reference system matches the PERSON argument to a PERSON mention in the snippet.
containsQuote	True if the snippet contains a quotation according to quotation-detector model.
matchEqTermXdcId	True if an equivalent term in the query has a cross-document co-reference id equal to a mention in the snippet.

Table 4.12: Sample of features used by the PERSON model

For the binary classifiers that constituted our argument and template models, we experimented with a number of different algorithms, including MaxEnt, averaged perceptron (Freund and Schapire 1999), support vector machines (Cristianini and Shawe-Taylor 2000) and passive aggressive models (Crammer *et al.* 2006). Based on this investigation, we opted for averaged perceptron models in all cases, due to their ability to converge quickly and provide superior results with the small data sets we were using. Also, the averaged perceptron is in the class of large-margin techniques that can deal with noisy, inseparable data, such as the feature-based representations of natural text used by our Distillation System. Finally, the algorithm, like many support-vector-based methods, is amenable to the use of Mercer kernels, which allow for efficient computation during training and classification. In our case, in order to provide effective exploration of the space of feature conjunctions, we used polynomial kernels of varying degrees (as small as two and as large as six). To give the reader a flavor for the features we would use, Table 4.12 shows a selection of features we used for our PERSON argument model.

4.4.3.4.5. Challenges

In spite of the averaged perceptron algorithm’s ability to deal with non-separable data, our biggest challenge was producing high quality annotated data. For example, when annotating data for the EVENT argument model it was often not trivial to determine whether a sentence explicitly mentioned an event. The following example illustrates this problem. A newswire article was describing a particular escape of four inmates from a Greek prison. One of the sentences considered for annotation was the following:

The police said that the two to escape with the help of accomplices they rented the helicopter from the town of Antiphonitis Sahelian Kwsmas in the south of the country.

This sentence did, indeed, refer to the particular Greek prison-break incident that was the centerpiece of the article, and even mentioned a town in Greece, but because at least two levels of inference including the use of world knowledge were required to arrive at that conclusion, we deemed this sentence a negative example, for it did not constitute an *explicit* mention of the event. Such cases were tricky, and abounded in our event/topic data.

We also encountered many cases of inconsistent annotation. In such cases, however, we were able to use our models themselves to catch the errors. Normally, a model can learn its training set quite well, and so we tested our models on their training sets and observed their errors. In many cases, the errors were instances of inconsistent annotation that we would correct, thereby improving performance on the model's held out development test set.

4.4.3.5. Results

We now compare the statistical snippet detection stage and a finely-tuned rule-based heuristic stage, under the assumption that the remaining components of the pipeline are identical.

4.4.3.5.1. Materials and Methods

Data for the experiments was obtained by constructing queries for each template, retrieving documents from the Gale corpus, and manually annotating these documents. Table 4.13 shows the number of training, devtest, and test examples for the templates used to produce plots in this section. The split was 80–10–10 based on the document of origin of the annotated data, which accounts for Training and devtest data were used to train and to develop features for the statistical models. Both systems were evaluated on the test data.

Template	T01	T03	T04	T09	T10	T12	T16
Training	4942	806	4875	3801	862	693	2444
DevTest	597	93	300	481	119	109	608
Test	806	59	450	581	76	171	308

Table 4.13: Data set sizes for the templates in the experiments

Both statistical and heuristic models operate by first computing a score and then selecting answers whose score exceeds a predefined threshold. This threshold determines the operating point of the model, which can be characterized in different ways. Here, we use two different characterization of the operating point: the first consists of computing the F-deficit, defined as $1 - F_1$, where F_1 is the traditional F-measure of information retrieval. The second is by means of a pair of probabilities: probability of missing $p_m(T)$ (i.e., probability of type II error), and probability of false alarm $p_f(T)$ (i.e., probability of type I error). The operating point of the model can then be represented as the point $[p_f(T), p_m(T)]$ in the $[0,1] \times [0,1]$ square. The operating point of the model corresponding to a specific value of the threshold can be estimated from the data, for example, by means of a test set separate from the training and development sets used to infer and tune the model.

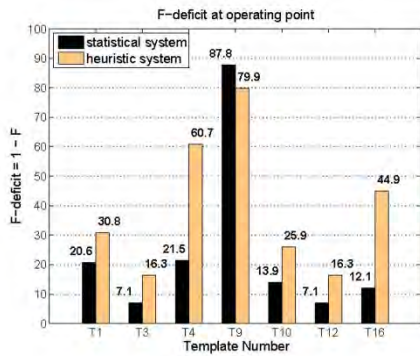


Figure 4.13: Comparison of statistical system and heuristic system on seven templates, in terms of F-deficit, defined as $1 - F_1$

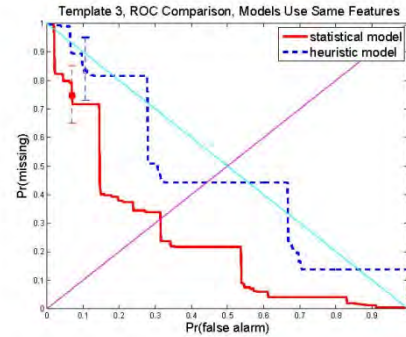


Figure 4.14: ROC curves of a heuristic and statistical systems for Template 3. The statistical system was trained with the same features used in the rule-based heuristic system.

In Figure 4.13 we provide an overall assessment of our methodology by comparing the F-deficit values across several templates. In the remaining experiments, we contrast the behavior of the two systems on individual templates by analyzing the dependence of $[p_f(T), p_m(T)]$ on the choice of the threshold. Given the finite nature of the test set, the set of points $\{[p_f(T), p_m(T)]\}_{T \in \mathcal{R}} \leftarrow$ is finite, and the curve obtained by linearly interpolating these points is the *empirical Receiver Operating Characteristic (ROC)* of the model.⁸ ROC curves are related to Precision-Recall curves (Davis and Goadrich 2006), and are better suited to assess the performance of models when the training and test data is not selected to reflect the proportion of positive and negative examples in the underlying distribution.

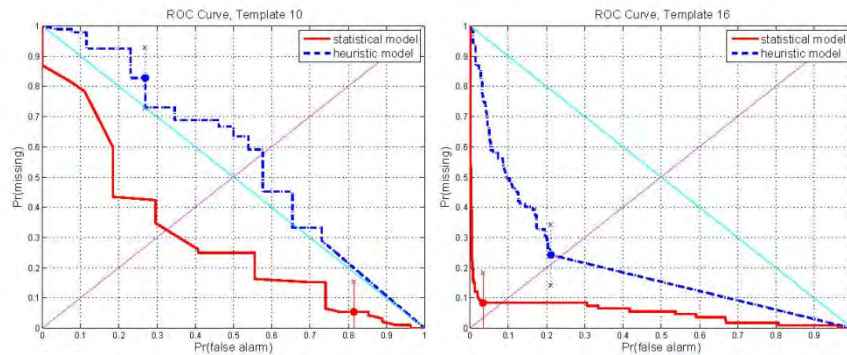


Figure 4.15: Typical comparison between statistical and heuristic models: the statistical ROC curve dominates the heuristic curve.

⁸ The points on these straight segments are achieved by flipping an appropriately biased coin to randomly select the thresholds of one of the two extreme points.

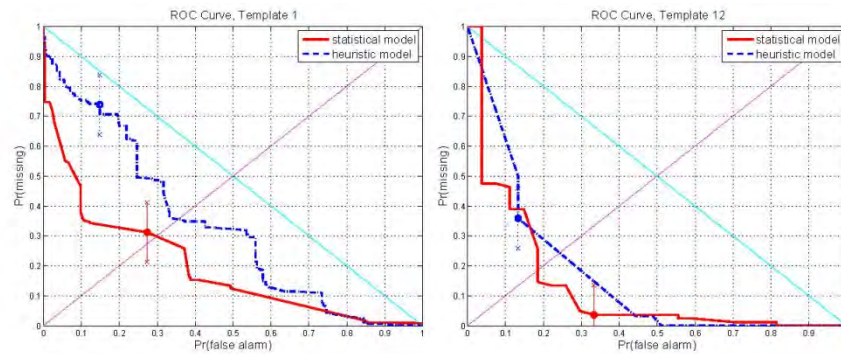


Figure 4.16: ROC curves where neither system uniformly dominates the other. On the left, the statistical system operates at a better point than the rule-based system; on the right, each system operates at a point where it locally dominates the other.

Figure 4.14, Figure 4.15, and Figure 4.16 show both the ROC curves obtained from the statistical model (solid line) and the heuristic model (dashed line), for a single template. Lower curves are better. Each plot also shows the equal-probability line (from $[0, 0]$ to $[1, 1]$) and the line describing a classifier that makes a decision by flipping a biased coin (from $[0, 1]$ to $[1, 0]$). The point corresponding to the threshold used in the system is marked on each ROC curve by a “o” mark crossed by a vertical bar.

4.4.3.5.2. Results and Discussion

Figure 4.13 shows the F-deficit values of the heuristic system and the statistical system at their actual operating point, on seven templates. In most cases, the statistical system reduces the deficit by half or more. Template 4 (provide information on ORGANIZATION), where the rule-based system yielded an F-deficit of above 60, the statistical system operates just above 20. On template 16 (describe attacks in LOCATION), the deficit reduction displayed by the statistical system is almost four fold. The exception is Template 9 (how did COUNTRY react to EVENT), where both systems have very poor performance, and the statistical system is inferior to the heuristic system.

The experiment illustrated in Figure 4.14 compares a statistical system and its corresponding rule-based system on Template 3 (provide information on PERSON), where both systems rely on the same 15 features.⁹ Recall that all the rule-based heuristic systems described in this section are carefully designed and developed to perform the Snippet Detection task, and are not simple systems intended just for seeding statistical models. The figure shows how the statistical model is uniformly better than the heuristic system: for each value of the probability of false alarm, it yields a smaller probability of miss than the heuristic system. Theorem 3.2 of (Davis and Goadrich 2006) ensures that the same dominance yields in Precision-Recall space, and hence in F_-measure space for every value of $\beta \in (0, \infty)$, every fixed dataset. While the ROC curve shapes are similar, the statistical system's shows finer-grained details. This illustrates the ability of the statistical

⁹ Only three of these features were used to construct the training set.

algorithm to perform local optimizations of the decision surface that would be difficult to capture with heuristics.

The remaining experiments compare heuristic and statistical systems constructed with different features. One of the characteristics of statistical systems is the ability to deal with classes of features, such as lexical features, that are not handled as well by heuristic systems. Figure 4.15 illustrates the typical results of the experiments: the statistical system dominates the heuristic system both on difficult templates (e.g., Template 10 — connection between EVENT/TOPIC and EVENT/TOPIC — where the heuristic model is not better than a biased coin) and on “simpler” templates (e.g., Template 16, where the heuristic system is substantially better than a biased coin).

Figure 4.16 illustrates the second most common behavior, where the statistical system is not uniformly better than the heuristic system. For the Template 1 (list facts about EVENT), the statistical system chooses a point closer to the equal-error line than the heuristic system; additionally, the point selected by the statistical system is superior to that selected by the heuristic system, in the sense that both operating points fall in a region where the statistical system is locally better than the heuristic system. For Template 12, the ROC curves are somewhat similar, and each system operates in a region where it is locally superior to the other.

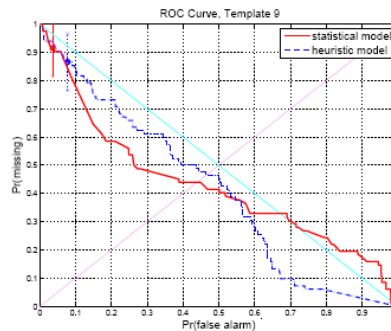


Figure 4.17: Example of a difficult template: while the heuristic, rule-based ROC curve mostly dominates the statistical ROC curve, both systems are dominated by a biased coin.

Figure 4.17 shows the case in which the statistical system is inferior to the heuristic system in terms of F-deficit: Template 9. Neither ROC curve dominates the other and neither convex hull contains the other curve. While the statistical system is somewhat better in the high-precision region of the curve, it is not substantially superior to a biased coin with the appropriate bias. At the operating point, the statistical system has even lower recall than the heuristic system. Both systems have a very high F-deficit across the entire ROC curve. We postulate that Template 9 requires more advanced features than those developed for the two systems, to better identify what constitutes a “reaction” to an event on the part of a country.

4.4.3.6. Summary

We believe we derived substantial benefits by first developing a heuristic system before moving on to a full statistical system. First, in developing the heuristic system we learned the problem well as researchers. Second, we identified features that would be useful when moving to the statistical system. Lastly, the heuristic system provided a meaningful baseline that proved to be an effective motivator when migrating to the fully statistical approach that we believed would surpass the performance of the heuristic system. As the empirical data have shown, this belief was correct. More generally, when presented with any new, complex task such as distillation, we advocate our approach.

4.4.4. Using Structural Semantic Information for Language Understanding Problems

Authors: Marjorie Freedman, Elizabeth Boschee, and John Graettinger

4.4.4.1. Introduction

Responding to the Distillation task involved identifying relevant, non-redundant responses to a query. For queries involving descriptions of topics or events, the system typically must first determine whether the candidate answer refers to the same topic or event as the query. Matching a topic or event can be trivial if the description consists of a single name or very specific term, but is a significant challenge when the description is lengthy or complex. One example of such an event is the query: “*List facts about [the widespread looting of Iraqi museums after the US invasion]*”. There are many difficulties in dealing with such a topic description.

- Some words will be less important than others. Here, ‘widespread’ is a relatively unimportant word and will be omitted from many relevant responses.
- Even seemingly important words can be omitted in relevant responses. “*Many Iraqi museums were looted, leaving Iraq without its cultural treasures*” omits the important information that this occurred after “the US invasion”.
- Sometimes on-topic information can be conveyed without using any of the words from a user’s specific topic formulation: “*Many works of art were stolen from Baghdad galleries in 2003*” is a relevant response that contains very little token overlap with the original query.
- The presence of the topic words in a sentence does not guarantee that a response will be relevant: “*Iraqi museums presented an exhibit on looting in Afghanistan after the US invasion*” is not a relevant response.
- The semantic role that words play in a question is important. A query about ‘*arrests by the policemen*’ is very different than a query about ‘*arrested policemen*’

The ability to find similar spans of text is necessary for tasks beyond question answering. Even during early distillation research, a related capability was necessary to

perform redundancy removal within a set of relevant snippets. In fact, the proposition trees described here were used in both BBN's answer finding and redundancy detection components. The task of identifying similar spans of text is also present in textual entailment, where systems must determine if a sentence is entailed by or contradicts a different sentence.

In this section, we will describe in detail the predicate-argument propositions extracted by SERIF, BBN's information extraction engine (Section 4.3.2), and how these propositions are integrated into a proposition tree (a kind of *text graph*) for systems to identify similarity between spans of text despite differences in terms and syntactic structure addressing the challenges outlined above. By incorporating semantic information into our measure of similarity we perform *semantic associative retrieval*, using semantic features (rather than lexical items) to detect similarity. We will describe experiments that explore the effectiveness of the semantic matching algorithm when compared with a word based approach and how the semantic algorithm can be adapted to Chinese and Arabic text.

We also describe how SERIF's propositions can be used in other contexts, for example to find the core *Who*, *What*, *When*, *Where*, and *Why* for a sentence with relatively simple heuristics, and as a basis for automatically learning patterns that trigger a relation.

4.4.4.2. Related Work

Tree edit distance metrics have been used successfully for factoid-question answering (e.g., Punyakanok *et al.* 2004; Emms 2006) and textual entailment (Kouylekov and Magnini 2005). Therefore, the appropriateness of tree edit distance is not surprising, though factoid question answering and entailment are significantly different than Distillation. Factoid-question answering and Distillation share a goal of responding to questions with relevant information. However, typically the response to a factoid-question is a single, short phrase, requiring high accuracy in finding a 1-best answer. In contrast, the typical Distillation query has many relevant answers, most of which will be a full clause or sentence; the distance metric must accurately rank the top n-candidate responses. As we discuss in Section 4.4.4.4, relevant answers frequently share only partial overlap in meaning with the question. For Distillation, unlike textual entailment, the distinction between related (but not entailing) phrases and even the distinction between entailment and contradiction does not affect a potential answer's correctness. As a result the system described here is designed to assess potential relevancy in situations where the edit distance is expected to be high.

4.4.4.3. Propositions

In Section 4.4.1.3.5, we describe how predicate-argument propositions are extracted from a syntactic parse as a part of the linguistic analysis performed by SERIF. As a representation of text content, the propositions extracted by SERIF occupy a middle ground between PropBank's (Palmer *et al.* 2005) semantic representation of verb argument structure and the purely syntactic predictions of SERIF's parser which is trained using the Penn Treebank (Marcus *et al.* 1993). The proposition finder normalizes

for certain common syntactic constructions, for example identifying the logical subject of a passive verb, but does not distinguish between verb-specific argument roles (e.g., *agent* versus *experiencer*). SERIF's proposition finder normalizes some prepositional phrase arguments, for example labeling certain arguments with location or temporal roles, but for other prepositional phrases, it simply attaches the argument with the preposition as the role label. SERIF's proposition finder considers both noun and verb predicates and attaches arguments to both. Currently, SERIF identifies propositions in English and Chinese. While we could develop proposition finding code for Arabic, we have not done so, partially due to the lower performance of the Arabic syntactic parser thus far.

Propositions have been a key component in many language understanding tasks including:

- BBN's event and relation models in both Chinese and English
- Distillation both to find relevant passages and identify redundant answer texts
- Initial experiments where a system "learns to read" concepts and relations (Boschee *et al.* 2008)

While in this section we focus on the use of propositions as a measure of similarity in a question answering framework, the propositions themselves are key components to BBN's pattern language described in Section 4.3.2. For example, when responding to the question *Find statements by [PERSON] on [TOPIC]*, the information that the person must be the logical subject of a statement-like verb is encoded with a proposition based pattern.

4.4.4.4. Proposition Trees for Identifying Similarity between a Query and Answer

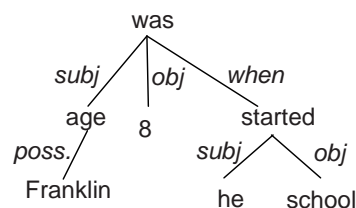


Figure 4.18: Proposition Tree for *Franklin's age was 8 when he started school*

Proposition Trees (Levit *et al.* 2007) combine three types of semantic information: SERIF's propositions, co-reference, and synonyms. The predicates and arguments identified by SERIF are linked in tree structure as illustrated in Figure 4.18. The structure of proposition-trees provides three contributions to identifying relevant, 'on-topic' responses.

- (1) Because the proposition finder has regularized the parses, we are able to recognize similarity across alternative syntactic manifestations of the same semantic concept (e.g., active versus passive constructions).

- (2) The distance of a word from the root in a proposition tree gives us a measure of the importance of the word in the sentence.
- (3) Edges of the proposition tree introduce a measure of proximity for long range dependencies.

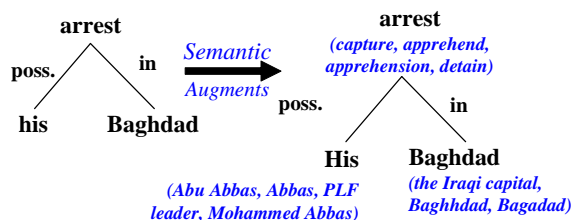


Figure 4.19: Synonym and Co-reference Augmentation for Proposition Trees

As we saw in the examples in the introduction, in addition to structural changes in a sentence, a distiller must be able to account for changes in the lexical items of a sentence. To account for such changes we augment nodes in the tree with synonyms (from WordNet) and co-reference (both for names, pronouns and noun phrases within a document, and for spelling variants and aliases across the full corpus.) Figure 4.19 illustrates the proposition tree before and after being augmented with synonym and co-reference information.

Figure 4.20 shows three proposition trees. The sentences represented by the trees are quite different (a) is a question; (b) is a noun phrase; and (c) is a complete sentence. The lexical items are also quite different, but because the proposition trees normalize the syntactic structure and augment the nodes with co-reference (e.g., British = UK) and synonyms (attack = strike), the distiller correctly recognizes that all three phrases are similar.

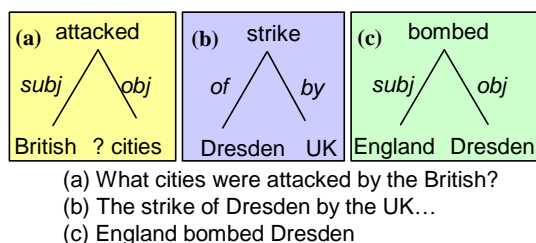


Figure 4.20: Proposition Trees for Three Text Spans

In addition to matching ‘full trees’ as seen in Figure 4.20, our system can be configured to match only sets of edges or only sets of nodes as shown in Figure 4.21. Such structures are still augmented with synonyms and co-reference, and in the case of edge matching, preserve the role labels and some structural information. In practice we typically combine different levels of matching. When working with the output of machine translation we have frequently found that the combination of edge and node matches is more robust than full match given the errors present in such text.

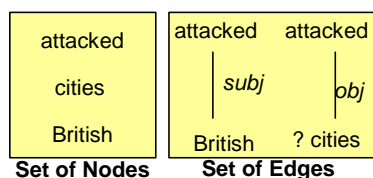


Figure 4.21: Sets of Nodes and Edges

4.4.4.5. Contributions of Synonyms and Structure Over English Text

One of the challenges of Distillation has been developing automatic metrics that correlate with the full distillation task. In Section 4.6.2.4, we describe evaluation of proposition tree matching in the context of the full GALE evaluation. Here we have chosen an automatic evaluation route that measures how often the ‘top’ match in a relevant document is a relevant answer.

Of the 17 Distillation templates, *Template 1: List facts about [EVENT]*, is the template that relies the most on a distiller’s ability to identify similar and related text. To evaluate proposition trees, we have taken documents which were annotated with answers to Template 1 queries and measured how frequently the sentence that was identified as the highest match was in fact a relevant answer to the query.

There are many aspects of distillation that this does not measure; notably, the need to identify all relevant information, and the need to filter through irrelevant documents. However, the relative simplicity of the measure allows us to run experiments in several configurations and thus empirically measure the impact of the various components of the proposition trees.

One reason we have not attempted to evaluate proposition trees as a similarity metric identifying all relevant answers in the document is that relevant answers do not need to overlap in meaning with the query. For example, given the query *List Facts about [the 2008 presidential election]* the snippet ‘*McCain and Obama held a debate*’, would be a relevant answer, but we would not expect the proposition tree algorithm to identify it as ‘similar to the query’.¹⁰ However, we would assume that any text which is similar to the query (e.g., *Before the US elects a new president in 2008...*) would be a relevant answer. As a result, we can assume that if the system selects an answer as being similar and that snippet is not relevant, then the system is incorrect in its judgment of similarity.

In our full Distillation System, we use heuristics that combine the three different approaches to similarity scores (node, edge, and full-match) for a sentence with the similarity scores of neighboring sentences to recover some of the relevant snippets that would otherwise be missed. While we have not performed an internal evaluation of the full approach, BBN’s system was the top performing system for five out six of the query types which included a *topic* or *event* slot including *Template 1: List Facts about [EVENT]*, for which all of BBN’s answers were the result of the semantic matching approach described here.

¹⁰ In Distillation section 4.4.4 we discuss techniques for expanding our measure of similarity to account for context outside of the answer snippet and thus identify such answers.

Our experiments compare edge match, node match, and node match without synonyms.¹¹ The experiment is intended to tease apart the relative importance of the different types of semantic information involved in our similarity metric. As discussed above, we measure only precision (i.e., in the form of how often the top-ranked sentence in a document was judged as relevant to a query). Given this experiment where only the top ranked answer was evaluated, we found that co-reference expansion was not critical, but if the task had been more recall oriented (e.g., identify all of the relevant snippets), co-reference might have played a larger factor. We did not measure full match in this set of experiments. Our intuition is that this setup would not have favored full match over edge match. Full match allows the system to rank highly those answers which capture the full meaning and structure of the query. These can be considered the ‘best’ matches. However, frequently a relevant answer may contain only portions of the query, and consequently answers where full match does not fire are also likely to be annotated as relevant. Because our answer key includes all relevant answers (with no measure of which answers are ‘better’), full match would have provided little benefit over edge match.

Matching Strategy	Correct	
	#	%
English Edge Match	205	88.0 %
English Node Matching	196	84.1 %
English Node Matching, without WordNet	184	79.0 %

Table 4.14: Number and Percent Correct Using Different Matching Strategies in English

Table 4.14 presents the results of applying three matching strategies to 233 query-document pairs.¹² In the two right-most columns, we include the number and percentage of the time each algorithm’s highest ranked snippet was a relevant snippet. Identifying relevant text with structure (edge matching) outperforms a strategy that used simply the lexical items augmented with synonyms. Furthermore, in the last row we see that augmenting the trees with synonyms outperforms a matching strategy that only makes use of the lexical items of the query.

4.4.4.6. Proposition Trees in Arabic and Chinese

As we describe in Section 4.6.2.4 we believe that distillation performance could be improved by performing answer identification in the source language. As a step towards a full source language Distillation System, we have implemented a version of the matching (semantic associative retrieval) algorithm in both Chinese and Arabic. Levit *et al.* (2007) describe proposition trees used over English text. SERIF is a multi-lingual engine and produces normalized proposition sets in Chinese (and English) and syntactic parses in English, Chinese and Arabic. Because propositions were not available in Arabic, we implemented an alternative method of generating proposition trees in which the trees are produced directly from the syntactic parse (without the intermediate step of

¹¹ In this setting, if there is no ‘edge match’ in a document, we choose the best ‘node match’.

¹² The query-document pairs were generated by selecting all documents with at least one relevant snippet for the LDC released Template 1 queries.

propositions). This transformation captures some but not all of the effect of the proposition finder: it uses a small number of heuristics (e.g., for prepositional phrases transforming the preposition into the role) to create the proposition trees. We use the same evaluation methodology described in the previous section. Table 4.15 Table 4.16 include results over 233 English query document pairs (from 21 queries), 143 Chinese query document pairs (from 16 queries), and 230 Arabic query-document pairs (from 17 queries).

Matching Strategy	Correct	
	#	%
English Edge Match- Propositions	205	88.0 %
English Edge Match- Parse	199	85.4 %
English Node Matching	196	84.1 %
Chinese Edge Match- Propositions	123	86.0%
Chinese Edge Match- Parse	123	86.0%
Chinese Node Match	122	85.3%
Arabic Edge Match- Parse	210	91.3%
Arabic Node Match	209	90.9%

Table 4.15: Comparison of Matching Strategies in English, Arabic, and Chinese

Matching Strategy	Correct	
	#	%
Chinese Match- Reference	123	86.0%
Chinese Match- Web	120	83.9%
Chinese Match- Pre GALE	93	65.0%
Arabic Match- Reference	210	91.3%
Arabic Match- Web	195	84.8%
Arabic Match- Pre GALE	182	79.1

Table 4.16: Degradation of Matching as a Result of Translation Quality

Experiment 1: Here we compare performance of ‘edge matching’ using propositions and/or parses in English, Chinese and Arabic with node matching in all three languages. For the purposes of this experiment, we use reference translations of the queries provided by the LDC. Results between languages are not directly comparable as we are using different queries and different documents. We have not yet included synonym expansion in Arabic or Chinese. Results suggest that structural matching can help in all three languages, although the effect is less strong in Arabic and Chinese, perhaps due to lower parsing accuracy in both languages. While in English, using the propositions is clearly an improvement over using a parse-based match, this is not true in Chinese. Potentially, improvements to the normalization process in the Chinese proposition finder could improve accuracy. Other improvements (to both node and edge matching) could come from incorporating Chinese and Arabic synonyms. The lack of synonyms may cause more degradation to edge match (which requires the match of two lexical items) than to node match.

Experiment 2: In a second experiment we examine the effect of translation quality on edge matching in Arabic and Chinese. Because distillation queries will be in English, translating the query into the source language is necessary to identify answers in the source. In this experiment, we used three translations; (1) a reference translation, (2) a web-based machine translation, and (3) a pre-GALE translation system developed at

BBN.¹³ In Table 4.16 we see that translation quality is important and that the accuracy of the proposition tree matching degrades rapidly as machine translation quality degrades. While unmeasured, we could imagine a similar degradation in performance occurs when matching proposition trees over machine translated material in English.

4.4.4.7. Other Applications of Propositions and Proposition Trees

Propositions and proposition trees are used for many other language understanding tasks as well. For instance, Boschee, *et al.* (2005) describe a BBN system that learns patterns for relationships from a few seed examples. To do this, the system iteratively searches a collection of documents to find sentences where instances of a known relations are expressed (e.g., for the relation *invent(Thomas Edison, light bulb)*, the system first finds instances such as “*Thomas Edison’s patent for the light bulb*”), and then induces patterns over textual features found in those instances, and finally repeats the cycle by applying the generated patterns to find additional instances followed by inducing more patterns from those instances, etc.

This research was built on prior work that took a similar approach but relied solely on surface-level features, i.e., tokens or parts-of-speech (e.g., Ravichandran and Hovy 2002). In our work, the system incorporates deeper, structural features based on proposition tree (and dependency tree) patterns. Initial results showed specifically that patterns based on predicate-argument propositions outperformed surface rules in both recall and precision. (Patterns based on dependency trees were more accurate than both, but fired far less frequently.) The following table shows the results from the first iteration of the system learning to recognize the relation *invent*.

	Correct	Incorrect	Percent Correct
Dependency Trees	75	10	88.2%
Propositions	315	86	78.6%
Surface Structure	222	68	76.6%

Table 4.17: first iteration of the system learning to recognize the relation *invent*

For this application, the primary strength of proposition-based patterns is their ability to generalize well across complex sentence structures. For instance, it is trivial for a proposition-based pattern to identify the sentence “*Trevithick also developed steam engines for use in mines and invented a steam threshing machine*” as an instance of an *invent* relation. For an example like that, it would be difficult to automatically generate surface-level patterns (as in prior work) that could reliably detect relations like this without introducing too many false positives.

4.4.4.8. Conclusion and Future Work

We have shown how SERIF’s propositions serve as a flexible representation of the logical predicate-argument structure of a sentence. The generic nature of the propositions has made them an invaluable tool for several of our language understanding components

¹³ English to Arabic and English to Chinese translation has not been a focus for GALE.

including distillation and additionally in recent work in learning patterns for entity/relation extraction.

While the propositions are language independent and can be effective in multiple languages, accuracy degrades as the quality of the text degrades. This presents a challenge for advancing such techniques in Distillation Systems where work is done in a cross-lingual setting. Propositions potentially provide a language model of English, Arabic, and Chinese; proposition extraction therefore may provide a semantic language model to contribute to more accurate transcription and translation

We believe that SERIF's propositions will continue to be useful in a variety of new language understanding tasks. If the proposition trees were augmented to allow the detection of contradiction, as well as, similarity, they could serve as an effective starting place for a system that performed the Rich Textual Entailment (RTE) task (Dagan *et al.* 2006).

4.4.5. Improving Answer Identification by Merging Distiller Responses

Authors: Jeff Ma, Elizabeth Boschee, and Marjorie Freedman

4.4.5.1. Introduction

Many different approaches to open-domain question-answering for complex questions (ones that do not have a single, short answer) have been explored, both in GALE Distillation and elsewhere. These approaches demonstrate diverse strengths and weaknesses; thus, a system which combines responses from different methods could improve performance beyond any individual algorithm.

We used a system combination approach to participate in the Distillation evaluation of the GALE program, where our final submission was generated by merging the results of two distinct systems. We show that the combined system outperforms both of the individual component systems. Results are presented for user queries formulated according to the GALE Distillation query templates (for example, *Describe the relationship of X and Y* or *How did country A react to event B?*), but the principles generalize easily to any class of questions having multiple partial answers.

4.4.5.2. Overall Approach

The two component question answering systems ("SYS1" and "SYS2") operate over the same corpus but ingest documents and produce responses independently. During response generation, each system selects a set of excerpts from documents ("snippets") that it believes to be relevant responses to the query. One system (provided by BBN) uses a linguistically-informed pattern recognizer to select the relevant answers; the other (provided by Lymba) selects snippets based on theorem proving.¹⁴ Given input from both systems, the Merger re-ranks and merges the two sets of snippets into a single list. To produce a final response, another module condenses the re-ranked list of snippets,

¹⁴ <http://lymba.com/>

removing redundancy and shortening the final response to what it decides is the ideal length (typically shorter than the full list of snippets). Here, we will focus solely on the Merger and will compare results of a fixed length.

Both systems generate scores for their snippets, but the scores are not comparable across systems and cannot be used directly to re-rank the list of combined snippets. The easiest way to combine snippets is to normalize the scores for each system and then use those scores to merge and re-rank the snippets. For our baseline, we assign a normalized score to each snippet by dividing its system generated score by the sum of all snippet scores generated for the same query by the same system. We consider this normalized score to be the snippet's posterior. Using these scores is one simple choice for merging.¹⁵ However, more complex techniques should prove more effective. We explore methods for estimating confidences for snippets, which we will use to re-rank snippet output for each system and to enable more effective merging.

4.4.5.3. Related Work

System combination techniques have been explored in natural language technology, e.g., speech recognition (Fiscus 1997, Ma and Matsoukas 2005), machine translation (Rosti and Ayan 2007; Hillard and Hoffmeister 2007), and information retrieval (Bartell *et al.* 1994; Si and Callan 2005).

In speech or MT systems the goal is to generate a single, potentially new, hypothesis by selecting words from a lattice of hypotheses produced by multiple systems. The successful use of confidences in speech and MT provides evidence that confidence-based merging of multiple independently ranked snippet lists may succeed.

System combination for factoid question answering (QA) requires finding the one-best answer, rather than accurately ranking multiple snippet lists. Merging multiple system outputs for lists requires systems to find specific, non-overlapping answers. Typically voting schemes can be a reasonable baseline in factoid/list QA.

Unlike speech, MT and factoid/list QA, information retrieval system must provide ranked lists of answers (whole documents/passages). However, much work here relies on assumptions that apply better to document retrieval than a task like Distillation. For instance, Si and Callan (2005) use a centralized retrieval algorithm to assess the quality of component retrievers (thereby avoiding the need for a manually-generated training corpus), but no such baseline algorithms are available for the more complex Distillation task, which cannot be approximated by term overlap with the query. Other IR efforts concentrate on systems designed to combine a large number of diverse components or algorithms (rather than just two) or systems that operate over multiple distinct collections of data.

By contrast to those efforts, this section reports on merging ranked lists of snippets arising from two highly distinct ranking algorithms. The key to our effort has been estimating and weighing confidence in each system's ranking based on tens of complex

¹⁵ This approach is more effective when the two systems produce a similar number of snippets. When responses for SYS1 and SYS2 are of very different lengths it may be misleading; however, as a baseline for comparison, it is a reasonable starting place (given that the systems do not produce any directly comparable confidence measures).

and rich features from the component distillation engines, moving beyond the word-based features used in IR system combination

4.4.5.4. Training Data

Annotated data for Distillation was provided by LDC; BBN also annotated data. To generate a set of correct responses for a training query, an annotator looked at potentially relevant documents (usually between 100 and 200 per query) and identified relevant responses. For instance, in response to the query *How were Mozart and Haydn related?*, an annotator reading a biography of Haydn might mark as relevant the sentence “*In 1781, Haydn met Wolfgang Amadeus Mozart in Vienna, and the two immediately became close friends.*” Such annotation is not comprehensive (since annotators cannot read every document in the corpus), but is sufficient to serve as an approximate reference. For the results reported here, we used 311 Distillation training queries, with responses generated from the TDT4 and TDT5 corpora (the official development data for the GALE Distillation Evaluation).¹⁶

To generate training data for system combination, we asked both systems to produce responses for these 311 queries. SYS1 generated 85,095 snippets; SYS2 generated 51,100 snippets. Queries where either system produced a null response were discarded, leaving 294 queries with responses from both systems.

We split the queries into two subsets, a development (Dev) set — 80% of the data — and a validation (Val) set — 20% of the data, selected randomly but attempting to maintain an 80/20 division of both queries and snippets. These data sets were used for all experiments reported here.

4.4.5.5. Metrics

4.4.5.5.1. Automatic Metrics

The Distillation Evaluation uses human judgments to evaluate performance by judging the correctness of each individual “nugget” of information in a response. Since such human judgments are extremely labor-intensive and moreover must be redone whenever system output changes, we wanted a metric that automatically measures system performance. This metric should correlate well with human judgments, to guarantee that improvements observed in this metric can also be observed in human judgments.

One automatic scoring metric is derived from “ROUGE” (Recall-Oriented Understudy for Gisting Evaluation), which has been used to automatically evaluate text summarization systems by comparing n -gram overlap between hypotheses and references (Lin and Hovy 2003; Lin 2004). In Distillation, typically there are multiple reference snippets for each hypothesis snippet. We therefore derive a new metric “ROT”³, (3 “Rouge” and “rot” are the French and German words for “red”, respectively) where for

¹⁶ For these experiments we used queries from Distillation templates 1, 3, 4, 5, 6, 8, 9, 15, and 16. (More information about the Distillation templates can be found in section 4.2.1.) These are the templates for which annotation was available in sufficient quantity.

each hypothesis (h), $ROT(h)$ is generated by calculating the average $ROUGE$ - n score ($n \in [1, 2, 3, 4]$) against each reference (r_i), and choosing the highest score.

Like ROUGE, ROT is recall-oriented: long snippets tend to have high ROT scores, but do not necessarily have better quality. In machine translation, the precision-oriented BLEU metric (Papineni, *et al.* 2002) deals with an analogous problem by penalizing short translations. We reverse this method here to penalize long snippets according to the following equation,

$$LP(h, r) = \begin{cases} 1, & \text{if } (\text{len}(h) \leq \text{len}(r)) \\ e^{-\frac{\text{len}(h)}{\text{len}(r)}}, & \text{if } (\text{len}(h) > \text{len}(r)) \end{cases}$$

where $\text{len}(\cdot)$ denotes the length of a hypothesis or a reference in terms of number of words. Applying this penalty on the ROT score, we have a penalized score, “PROT”,

$$PROT(h) = \underset{1 \leq i \leq I}{\operatorname{argmax}} \{LP(h, r_i)\}$$

We show a comparison of the effectiveness of ROT and PROT in Section 4.4.5.7.

4.4.5.5.2. Human Snippet Judgments

One problem with ROT and PROT is that there will likely be system-generated snippets that are relevant to the query but are not similar to the set of reference snippets, since the annotation is not comprehensive. A metric closer to full human evaluation is desirable to determine whether system combination is effective. Comprehensive nugget-based human scoring, as in the Distillation Evaluation, is clearly out of scope. We approximate the formal evaluation by manually assigning human judgments (1 = relevant, 0.5 = mediocre, or 0 = irrelevant) to each snippet produced by one or both of the systems⁴. (4 We were able to assign scores to all of SYS1's validation snippets, but only about one quarter of SYS2's validation snippets, due to time constraints on this labor-intensive process.) Note that the level of granularity here is relatively crude (many snippets are partially relevant, which ROT/PROT measures more accurately), and this metric also treats all snippets equally, regardless of length. It, therefore, provides a reasonable way to evaluate performance improvements, but should not be considered equivalent to a full, formal human evaluation of system performance.

4.4.5.5.3. Metric Evaluation

To roughly evaluate the quality of our new automatic metrics, we determined how well they correlate with our human snippet judgments, by computing the Pearson correlation coefficients between snippet-judgment scores and automatic ROT and PROT scores for SYS1's outputs on the Val set. As shown in Table 4.18, both have a relatively weak (though positive) correlation with snippet judgment scoring, which we believe to be a somewhat closer approximation of the manual evaluation. The PROT scores also correlate with human judgments slightly better than the ROT scores do:

	ROT	PROT
Pearson Correlation Coefficient	0.197	0.205

Table 4.18: Pearson correlation coefficients between the snippet-judgment scores and the automatic scores

As shown, training models using ROT and PROT results in improvement measured by human snippet judgments, despite the relatively weak correlation of the metrics. Developing automatic metrics more tightly correlated with human judgments (or potentially just using a training corpus with more extensive references) would likely provide even greater performance increases, and is an interesting area for potential future work.

4.4.5.6. Snippet Confidence Estimation

4.4.5.6.1. Procedure

We have used the generalized linear model (GLM) (Hastie and Tibshirani 1990) in a variety of confidence estimation applications. It is expressed as:

$$E(Y) = \mu = g^{-1}(X \bullet \beta)$$

where $g(\mu)$ is a non-linear function that links the expected value of the response, $E(Y)$, and the linear predictor $X \bullet \beta$. μ is the mean of the distribution of the response Y . In our application Y is confidence. Since confidence scores are between 0 and 1, we choose $g(\mu)$ to be the “logit” function:

$$g(\mu) = \ln(\mu/(1-\mu))$$

The model parameter β is estimated by the iterative re-weighted least square method (McCullagh and Nelder 1989), given a set of input data X and output confidence Y .

To train a GLM to estimate confidence for system-generated snippets, we extract a set of snippet related features (i.e., produce input data X) and determine the relevance of snippets (i.e., produce output data Y).

To produce input data X , we create a set of snippet-related features by extracting the snippet related features in both systems' outputs. We extracted ten features for each snippet from SYS1 and six features for each snippet from SYS2. Some of these features are *number-of-chars*, *number-of-words*, *snippet-score* (generated by the component system) and *snippet-posterior* (a normalized version of *snippet-score*, as described in Section 4.4.5.2).¹⁷ Other system-specific features attempt to capture the system's basis for selecting a particular snippet. For instance, the snippet “*Haydn met Wolfgang Amadeus Mozart in Vienna*”, might have additional features *document_rank* = 0.34, *mention-confidence* = 1.0, *backoff_level* = 0 and *pattern_type* = *propositional-connection*.

¹⁷ We omitted the *number-of-chars* and *number-of-words* features in the final model training for SYS1 in the evaluation, because we found later that the performance on the Dev and Val sets was more consistent without those two features.

To produce output data Y , we have a set of human-selected reference snippets available for all queries in training. Using this data, we compute the ROT or PROT score for each snippet. If the ROT or PROT score is above a threshold, the snippet is considered relevant. If a snippet is defined as relevant, its target value (or Y) is set to 1; otherwise, to 0.

A threshold effectively divides the training data into two pieces: a set of “correct” snippets and a set of “incorrect” snippets. After some experimentation, using ROT, we found that the trained models generalize better when, rather than selecting a single arbitrary threshold, we compute an average confidence score for each snippet using a set of different threshold values. The average score:

$$avg_{cf}(snpt) = \sum_{t \in T} cf_t(snpt) / N_t$$

where T denotes a set of thresholds, N_t the number of thresholds in the set, and $cf_t(snpt)$ the confidence score estimated with threshold t . We used three threshold values, 0.1, 0.2 and 0.3, in all experiments, and trained confidence models for each system using development data.

Finally, all queries in the training data were from nine Distillation templates. The number and nature of the snippets produced for queries from each template varies greatly. We tried two training mechanisms: first, training a single joint confidence model for all data; and second, training nine separate confidence models for the nine templates. We found that the nine separate confidence models outperformed the one joint model, although a few templates suffered from over-fitting due to lack of training data (queries formulated according to certain templates tend to produce far fewer snippets). All experiments reported here use nine separately trained confidence models.

4.4.5.6.2. Results

Template ID	SYS1		SYS2	
	Dev	Val	Dev	Val
1	0.103	0.073	0.187	0.125
3	0.185	0.119	0.357	-0.141
4	0.122	0.060	0.267	0.133
5	0.167	0.026	0.227	0.069
6	0.086	-0.352	0.447	0.094
8	0.175	0.074	0.325	0.270
9	0.199	-0.338	0.329	-0.040
15	0.264	-0.265	0.178	0.084
16	0.113	0.044	0.128	0.028
Overall	0.164	0.049	0.263	0.128

Table 4.19: NCE scores for Dev and Val sets

We first look at an indirect measure of performance improvement, NCE (Normalized Cross Entropy). NCE is a common way to measure quality of confidence estimation (NIST 2000). The closer NCE is to 1.0, the better the quality of the confidence

estimation. Table 4.19 shows the NCE scores computed on both the Dev sets and the Val sets, using confidence models trained for both systems.

For many templates, the NCE scores are much lower on the Val sets than on the Dev sets. This is especially true for Templates 6, 9, and 15 for SYS1 and for Templates 3 and 9 for SYS2. These templates are those which have the smallest amount of training data (e.g., less than one-tenth of the data available for Template 1). We suspect that the models trained for these templates suffered from over-fitting during training.

In general, the gap between overall Dev NCE scores and overall Val NCE scores leads us to conclude that models trained on the Dev sets did not generalize very well to the Val sets. There are two possible causes for this behavior. First, there may not be enough training data. Second, the features used by the model may not generalize well across queries. NCE scores are in fact relatively low across the board — much smaller than 1.0, and also lower than typical NCE scores for word confidence estimation in STT (typically above 0.3). We, therefore, suspect the primary problem here is the quality and quantity of features. (Word confidence estimation models used for STT typically employ about 60 features, many more than for this task.)

Although the NCE scores are relatively low, using estimated confidence scores did improve the snippet ranking. To automatically compare two snippet rankings, we first cap the length of the output for each query at 500 words.¹⁸ We then compute the arithmetic average of ROT scores. If a re-ranking increases the average ROT, the re-ranking has successfully moved more correct snippets up rank. (All initial experiments were done with ROT.)

System	Ranked by	Average ROT scores	
		Dev	Val
SYS1	Posteriors	0.2843	0.2806
	Confidence	0.3751	0.3579
SYS2	Posteriors	0.1713	0.1818
	Confidence	0.2747	0.2794

Table 4.20: Average ROT scores before and after confidence-based re-ranking.

Table 4.20 compares the original snippet rankings (using the posteriors, which are proportional to the original system-generated scores) with the snippet re-rankings using the estimated confidences.

These comparisons show that re-ranking using confidence scores substantially increases the average ROT for each system on both the Dev and Val sets. The relative increases are about 25% for SYS1 and about 50% for SYS2. (The larger improvements for SYS2 are mainly due to that system's lower baseline scores.)

Figure 4.22 shows some representative examples of improved ranking on SYS1's output for a query in the validation set, *Tell me about the attempted British Airways plane crash?* Looking at the top ten snippets before and after re-ranking, we see improvements because correct snippets have been promoted (all three were originally ranked 45th or lower) and from the demotion of incorrect snippets. For this example, confidence-based

¹⁸ 500 words roughly corresponds to the desired response length for the Distillation go/no-go evaluation. In all comparison experiments reported in this paper, we cap the query output at 500 words.

re-ranking increased the number of correct, top ten snippets from three out of ten to six out of ten.

Query: Tell me about the attempted British Airways plane crash.

Correct snippets promoted into the top ten by re-ranking	Incorrect snippets demoted from the top ten by re-ranking	Correct snippets in the top ten before and after re-ranking
<p>-- A number of passengers are believed to have suffered relatively minor injuries as the aircraft made sudden maneuvers during the struggle.</p> <p>-- The man, described by passengers as tall, strong and hysterical, launched his assault on the cockpit during a British Airways flight from London to Nairobi.</p> <p>-- "Our crew are trained to deal with every situation and it was this training and their professionalism that ensured a quick resolution of the incident," Mike Street, the BA director of customer service, said in a statement.</p>	<p>-- More than 2.5 million passengers have since flown on BA's supersonic jets.</p> <p>-- He claims Virgin, which he created in the 1980s as an upstart challenger to BA, may be able to make the plane more profitable.</p> <p>-- Double said BA was talking to aviation museums interested in taking custody of the sleek jets.</p>	<p>-- A male passenger on a British Airways flight from London to Nairobi broke into the cockpit and tried to crash the plane Friday, sending it into a violent dive before the crew regained control, the airline and passengers said.</p> <p>-- But a British Airways spokesman said the crew was "in shock," and passengers making connections at Nairobi's Jomo Kenyatta International Airport said they thought the plane was going to crash.</p>

Figure 4.22: Examples of improved results from confidence-based re-ranking (SYS1)

We measured performance using the manual snippet judgments described in Section 4.4.5.5.2, by computing the arithmetic mean of snippet judgment scores on capped outputs (for the Val sets, since the manually-assigned scores do not exist for the Dev sets). Table 4.21 shows improvement in average snippet-judgment (SJ) score after confidence-based re-ranking. Here ROT does indeed correlate with simple human judgments, though the relative improvements on the average snippet-judgment score — about 7% for both systems — are smaller than the improvements observed on the average ROT scores — about 25% and 50% (shown in Table 4.21), due no doubt to the correlation being somewhat weak.

System	Ranked by	Average SJ Scores
SYS1	Posteriors	0.7704
	Confidence	0.8189
SYS2	Posteriors	0.6033
	Confidence	0.6439

Table 4.21: Average snippet-judgment scores before and after confidence-based re-ranking.

We also compared performance using ROT and the PROT on system SYS1. We trained new confidence models for SYS1, replacing the ROT metric with the PROT metric; then re-ranked the snippets using the confidence scores generated by the new

models. Table 4.22 shows this comparison in terms of the average snippet-judgment score; PROT outperforms ROT in terms of average snippet-judgment scores.

Metric	Ranked by	Average SJ Score
-	Posteriors	0.7704
ROT	Confidence	0.8189
PROT	Confidence	0.8398

Table 4.22: Comparison of the ROT and PROT metrics in terms of average snippet-judgment scores.

Given these results, we used PROT in our system merging work discussed in the next section.

4.4.5.7. System Merging

Given the positive results in the previous section, experiments on merging snippets from the two systems and re-ranking the snippets according to confidence scores began. Note that (1) the two systems do not typically perform equally well and should probably be weighted unequally and (2) that a snippet found by both systems is presumably more likely to be correct.

We applied a system-dependent scaling factor (W_{system}) to the estimated confidence scores and for each snippet *also* calculated an “overlap rate” with snippets produced by other systems. (The overlap rate for a snippet is the number of its tokens that also occur in snippets from other systems, divided by the total number of its tokens.) For each snippet, the *mrgSrc* score is defined as:

$$mrgSrc(snpt) = \alpha * W_{system} * confidence(snpt) + (1 - \alpha) * overlapRate(snpt)$$

Here, α is a system-independent factor that controls the relative weighting of the system-weighted confidence score and the overlap rate. Both W_{system} and α are in $[0, 1]$. The combiner re-ranks the snippets for each query according to their *mrgSrcs*. α and W_{system} are automatically optimized using the Dev sets from both systems, and tested using the combination of the Val sets from both systems. The best values for the merging parameters are found by linearly sampling the region of $[0, 1]$ with a sampling step of 0.1 for all parameters. For each template, we choose the parameter values that maximize PROT on the merged output.

We evaluate merging using both the original posterior scores as confidences and the estimated snippet confidence scores. Table 4.23 shows the results of both experiments in terms of average PROT for the individual component system responses and for the merged response. Percentage improvement attributed to merging is given as the relative improvement over the best individual system.

Comparing the merged response with individual system responses, one can see that either type of merging produces higher average PROT scores on both the Dev and the Val sets. Confidence-based merging produces slightly smaller relative improvements than

posterior-based merging because its baselines have already been improved by confidence-based re-ranking.

		Score used in merging	
		posteriors	confidence
On Dev set	SYS1	0.268	0.315
	SYS2	0.145	0.187
	Merged	0.314	0.347
	Relative Imprv.	17%	10%
On Val set	SYS1	0.238	0.253
	SYS2	0.141	0.169
	Merged	0.248	0.295
	Relative Imprv.	19%	17%

Table 4.23: Improvements from merging in terms of the average PROT score.

Comparing confidence-based merging with posterior-based merging, one can see that confidence-based merging provides higher overall PROT scores than posterior-based merging on both the Dev set (11% relative, improving from 0.314 to 0.347) and the Val set (4% relative, improving from 0.284 to 0.295). This verifies our initial intuition that confidence scores are more useful than posteriors in system merging. Measured on the Val set, the overall gain using both confidence estimation and system merging is 24% relative improvement over the best individual system.

4.4.5.8. Conclusion

We have presented a statistical approach for improving Distillation performance, and open domain question answering performance in general, using confidence estimation and system combination. We have presented a method for estimating snippet confidence, using trained GLMs and new automatic metrics designed for this task (ROT and PROT). System performance improved by re-ranking snippets according to confidence scores generated by these trained models. Moreover, when used as input to the final merging process, confidence scores improve the performance of the merger as well. The method we developed for statistically combining results from two component systems provides additional benefit for question answering performance; the final response is of higher quality than either individual system response. The combined result of both approaches (re-ranking based on confidence estimation and system combination) is of even higher quality. Desirable future directions include further work to improve both the automatic metrics and the confidence estimation procedure. The automatic metric used to train these models does not correlate very strongly with the simple human judgments we created. It would be interesting to explore metrics that could better approximate more complex evaluation of question-answering performance. The poor performance of confidence estimation in terms of the NCE metric is most likely explained by the shortage of good snippet-related features. Future work could include the development of more and better features for snippets, as well as, the creation of additional training data for the templates which are sparsely represented, potentially reducing overfitting. Improvements to the

automatic metrics and confidence estimation should also carry over to system merging, allowing for even further improvement over the already promising results presented here.

4.4.6. Identifying Conflicting Information in Texts

Authors: Marie-Catherine de Marneffe, Anna N. Rafferty, and Christopher Manning

4.4.6.1. Introduction

The primary goal of Distillation is to identify information relevant to a user's query. To do so, a system needs to gather pertinent and concise information about the topic in question. A crucial step in this process is to understand the relationships between pieces of text which are retrieved: to choose which information seems relevant, it matters to know whether text snippets are unrelated, equivalent, redundant, or whether they present information that conflicts. So far, little work has targeted the notion of "contradiction" in NLP tasks (Harabagiu *et al.* 2006), however as pointed out by Condoravdi *et al.* (2003), handling contradiction is necessary to achieve text understanding. We argue that it is also central to information analysis. It is particularly important for analysts to be made aware of conflicting factual claims and divergent viewpoints, which might reflect different sources, political leanings, or even disinformation. Consider the following texts:

- (1) Maitur Rehman, a 29-year-old Pakistani from Multan in Punjab, is reported to be the present amir of Jundullah. He had previously served in the Lashkare-Jhangvi, an anti-Shia terrorist organization.
- (2) Intelligence sources in the U.S. and Pakistan tell NBC News that Maitur Rehman is a low-level militant operating in South Waziristan.

If one wanted to know the status of Maitur Rehman, it would be appropriate to retrieve both passages, and let the user know that the information found diverges: if Maitur Rehman is an amir (a leader), then he is not a low level militant. The pieces of text contain conflicting information, and determining the exact status of Maitur Rehman would demand further analysis. Relevant information gathering can benefit from recognizing such contradictions in text.

We will analyze the nature of conflicting information that commonly appears in text, and describe a system to automatically identify these "contradictions". Information conflicts appear when facts diverge: a rising death toll for example, if detached from a time line, shows contradictory figures:

- (3) The explosion in Qana that killed more than 50 civilians has presented Israel with a dilemma.
- (4) An investigation into the strike in Qana found 78 confirmed dead thus far.

Conflicting opinions and reports are also very frequent in texts: one example is the pair of sentences in (1) and (2), another example is the following, where two different sources report conflicting data.

- (5) That police statement reinforced published reports that eyewitnesses said de Menezes had jumped over the turnstile at Stockwell subway station and was wearing a padded jacket, despite warm weather.
- (6) However, the documents leaked to ITV News suggest that Menezes, an electrician, walked casually into the subway station and was wearing a light denim jacket.

Distillation would benefit greatly from recognizing differences in opinions: it can indeed matter to a user that the two texts above differ in what information is embedded about de Menezes. We, therefore, aim at finding contradictions defined in a broad sense: two pieces of text are contradictory if they are extremely unlikely to be considered true simultaneously (de Marneffe *et al.* 2008). This definition captures the two cases of conflicting information that typically arise in text: divergent facts and different opinions.

We will describe a system which, given pairs of passages called text (T) and hypothesis (H), decides whether or not they are contradictory. A contradiction system is a key component for achieving relevant information gathering, and as such, could be integrated in a global system targeting distillation.

4.4.6.2. System Description

Our system to detect contradiction is an adaption of the Stanford RTE (Recognizing Textual Entailment) system (MacCartney *et al.* 2006). It follows the Stanford System multi-stage architecture. The first stage computes the linguistic representations containing information about the semantic content of the passages: the text and hypothesis are converted to typed dependency graphs produced by the Stanford parser (Klein and Manning, 2003; de Marneffe *et al.* 2006). The second stage provides an alignment between the graphs, consisting of a mapping from each node in the hypothesis to a unique node in the text or to null. Details about the scoring alignment measure and the search algorithm are presented by de Marneffe *et al.* (2007) and Chambers *et al.* (2007).

In the final stage, we extract contradiction features on which we apply logistic regression to classify the pair as contradictory or not. Feature weights are handset, guided by linguistic intuition. Contradiction features rely on mismatches between the text and the hypothesis. However, pairs of sentences which do not describe the same event, and thus cannot be contradictory to one another, could nonetheless contain mismatching information. An extra stage to filter non-co-referent events is, therefore, added before feature extraction. For example, in the following pair, it is necessary to recognize that the Johnstown Flood has nothing to do with a ferry sinking; otherwise conflicting death tolls result in labeling the pair a contradiction.

T: More than 2,000 people lost their lives in the devastating Johnstown Flood.

H: 100 or more people lost their lives in a ferry sinking.

This issue does not arise for textual entailment: elements in the hypothesis not supported by the text lead to non-entailment, regardless of whether the same event is described. For

contradiction, however, it is critical to filter unrelated sentences to avoid finding false evidence of contradiction when there is contrasting information about different events.

4.4.6.3. Filtering Non-Co-Referent Events

Right now the system uses a crude filter based on topicality. Assuming two sentences of comparable complexity, we hypothesize that modeling topicality could be used to assess whether the sentences describe the same event. The topicality score of a sentence is calculated as a normalized score across all aligned NPs. The text and hypothesis are topically related if either sentence score is above a tuned threshold. While filtering provides improvement in performance (6.8% in precision), some examples of non-co-referent events are still not filtered, such as:

T: Also Friday, five Iraqi soldiers were killed and nine wounded in a bombing, targeting their convoy near Beiji, 150 miles north of Baghdad.

H: Three Iraqi soldiers also died Saturday when their convoy was attacked by gunmen near Adhaim.

It seems that the real world frequency of events needs to be taken into account. In this case, attacks in Iraq are unfortunately frequent enough to assert that it is unlikely that the two sentences present mismatching information (i.e., different location) about the same event. But compare the following example:

T: Princess Diana died in Paris.

H: The car accident killing Princess Diana occurred in London.

The two sentences refer to one unique event, and the location mismatch renders them contradictory.

4.4.6.4. Contradiction Features

Mismatching information between sentences is often a good cue of non-entailment (Vanderwende *et al.* 2006), but it is not sufficient for contradiction detection, which requires more precise comprehension of the consequences of sentences. Some of the features used in the Stanford RTE system have been more precisely defined to only capture mismatches in similar contexts, instead of global mismatching. These features are described below.

Antonymy features. Aligned antonyms are a very good cue for contradiction. Our list of antonyms and contrasting words comes from WordNet, from which we extract words with direct antonymy links and expand the list by adding words from the same synset as the antonyms. We also use oppositional verbs from VerbOcean. We check whether an aligned pair of words appears in the list, as well as, checking for common antonym

prefixes (e.g., anti-, un-). The polarity of the context is used to determine if the antonyms create a contradiction.

Polarity features. Polarity difference between the text and hypothesis is often a good indicator of contradiction, provided there is a good alignment:

T: A closely divided Supreme Court said that juries, and not judges, must impose a death sentence.

H: The Supreme Court decided that only judges can impose the death sentence.

The polarity features capture the presence (or absence) of linguistic markers of negative polarity contexts. These markers are scoped such that words are considered negated if they have a negation dependency in the graph or are an explicit linguistic marker of negation (e.g., simple negation (not), downward-monotone quantifiers (no, few), or restricting prepositions). If one word is negated and the other is not, we may have a polarity difference. This difference is confirmed by checking that the words are not antonyms and that they lack unaligned prepositions or other context that suggests they do not refer to the same thing. In some cases, negations are propagated onto the governor, which allows one to see that *no bullet penetrated*, and *a bullet did not penetrate*, have the same polarity.

Number, date and time features. Numeric mismatches can indicate contradiction (see example (3–4) above). The numeric features recognize (mis-)matches between numbers, dates, and times. We normalize date and time expressions, and represent numbers as ranges. This includes expression matching (e.g., *over 100* and *200* is not a mismatch). Aligned numbers are marked as mismatches when they are incompatible and surrounding words match well, indicating the numbers refer to the same entity.

Structural features. These features aim to determine whether the syntactic structures of the text and hypothesis create contradictory statements. For example, we compare the subjects and objects for each aligned verb. If the subject in the text overlaps with the object in the hypothesis, we find evidence for a contradiction. Consider:

T: Jacques Santer succeeded Jacques Delors as president of the European Commission in 1995.

H: Delors succeeded Santer in the presidency of the European Commission.

In the text, the subject of *succeed* is *Jacques Santer* while in the hypothesis, *Santer* is the object of *succeed*, suggesting that the two sentences are incompatible.

Factivity features. The context in which a verb phrase is embedded may give rise to contradiction:

T: The bombers had not managed to enter the embassy.

H: The bombers entered the embassy.

Negation influences some factivity patterns: *Bill forgot to take his wallet* contradicts *Bill took his wallet*, while *Bill did not forget to take his wallet* does not contradict *Bill took his wallet*. For each text/hypothesis pair, we check the (grand)parent of the text word aligned to the hypothesis verb, and generate a feature based on its factivity class. Factivity classes are formed by clustering our expansion of the PARC lists of factive, implicative and nonfactive verbs (Nairn *et al.* 2006) according to how they create contradiction.

Modality features. Simple patterns of modal reasoning are captured by mapping the text and hypothesis to one of six modalities (*(not_)possible*, *(not_)actual*, *(not_)necessary*), according to the presence of predefined modality markers such as can or maybe. A feature is produced if the text/hypothesis modality pair gives rise to a contradiction. For instance, the following pair will be mapped to the contradiction judgment (*possible, not possible*):

T: The trial court may allow the prevailing party reasonable attorney fees as part of costs.

H: The prevailing party may not recover attorney fees.

Relational features. A large proportion of the RTE data is derived from information extraction tasks where the hypothesis captures a relation between elements in the text. Using Semgrep, a pattern matching language for dependency graphs, we find such relations and ensure that the arguments between the text and the hypothesis match. In the following example, we detect that Fernandez works for FEMA, and that because of the negation, a contradiction arises.

T: Fernandez, of FEMA, was on scene when Martin arrived at a FEMA base camp.

H: Fernandez doesn't work for FEMA.

Relational features provide accurate information, but are difficult to extend for broad coverage.

4.4.6.5. Typology of contradictions

The corpora we used to develop our system are the RTE datasets (Dagan *et al.* 2006; Bar-Haim *et al.* 2006; Giampiccolo *et al.* 2007) that we annotated for contradiction. We found that contradictions constitute approximately 10% of these corpora. Since the RTE datasets were constructed for textual inference, they might not reflect 'real life' contradictions. We therefore also collected contradictions 'in the wild.' The resulting corpus contains one hundred and thirty one contradictory pairs, and is available at

<http://nlp.stanford.edu/projects/contradiction>. About 40% of the pairs were taken from the GALE distillation data. We excluded repeated pairs. We, therefore, took 8 pairs out of the 45 “contradicting” pairs in the Phase-1 data, and 45 out of the 72 pairs judged contradictory in the Phase-2 data.

Analyzing the data, we find two primary categories of contradiction: (1) those occurring via antonymy, negation, and date/number mismatch, and (2) contradictions arising from the use of factive or modal words, structural and subtle lexical contrasts, as well as, world knowledge (WK). Category (1) contradictions are more often surface contradictions, which are relatively easy to detect, as in the death toll example (3–4)) or in the de Menezes example (5–6). The contradictions in the second category are more difficult to find automatically: they involve lexical and structural discrepancies, as well as, inconsistency via world knowledge. In (1–2) for instance, the meaning of *amir* is crucial for detecting the contradiction. In the following pair, one needs to have some knowledge about head companies and branches:

- (7) Microsoft Israel, one of the first branches outside the USA, was founded in 1989.
- (8) Microsoft was established in 1989.

4.4.6.6. Results and Discussion

	Precision	Recall
RTE1 dev1	70.37	40.43
RTE1 dev2	72.41	38.18
RTE2 dev	64.00	28.83
RTE3 dev	61.90	31.71
RTE1 test	42.22	26.21
RTE3 test	22.95	19.44
Avg. RTE3 test	10.72	11.69

Table 4.24: Precision and recall figures for contradiction detection. ‘Avg. RTE3 test’ refers to mean performance of the 12 submissions to the RTE3 Pilot.

Our contradiction detection system was developed on all datasets listed in the first part of Table 4.24. As test sets, we used RTE1 test, as well as, the RTE3 test independently annotated by NIST (Voorhees 2008). We focused on attaining high precision. In a real world setting, it is likely that the contradiction rate is extremely low; rather than overwhelming true positives with false positives, rendering the system impractical, we mark contradictions conservatively. We found reasonable inter-annotator agreement between NIST and our post-hoc annotation of RTE3 test ($\kappa = 0.81$), showing that, even with limited context, humans tend to agree on contradictions.¹⁹ The results on the test sets show that performance drops on new data, highlighting the difficulty in generalizing from a small corpus of positive contradiction examples, as well as, underlining the complexity of building a broad coverage system. This drop in accuracy

¹⁹ This stands in contrast with the low inter-annotator agreement reported by Sanchez-Graillet and Poesio (2007) for contradictions in protein-protein interactions. The only hypothesis we have to explain this contrast is the difficulty of scientific material.

on the test sets is greater than that of many RTE systems, suggesting that generalizing for contradiction is more difficult than for entailment. Particularly when addressing contradictions that require lexical and world knowledge, we are only able to add coverage in a piecemeal fashion, resulting in improved performance on the development sets, but only small gains for the test sets.

Type	RTE3 dev	RTE3 test
1 Antonym	25.0 (3/12)	42.9 (3/7)
Negation	71.4 (5/7)	60.0 (3/5)
Numeric	71.4 (5/7)	28.6 (2/7)
2 Factive/Modal	25.0 (1/4)	10.0 (1/10)
Structure	46.2 (6/13)	21.1 (4/19)
Lexical	13.3 (2/15)	0.0 (0/12)
WK	18.2 (4/22)	8.3 (1/12)

Table 4.25: Recall by contradiction type.

Thus, as shown in Table 4.25, we achieve 13.3% recall on lexical contradictions in RTE3 dev, but are unable to identify any such contradictions in RTE3 test. Additionally, we found that the precision of category (2) features was less than that of category (1) features. Structural features, for example, caused us to tag 36 non-contradictions as contradictions in RTE3 test, over 75% of the precision errors. Despite these issues, we achieve much higher precision and recall than the average submission to the RTE3 Pilot task on detecting contradictions, as shown in the last two lines of Table 4.24. In the RTE3 Pilot task, systems made a three-way decision as to whether pairs of sentences were entailed, contradictory, or neither (Voorhees 2008).

One significant issue in contradiction detection is lack of feature generalization. This problem is especially apparent for items in category (2) requiring lexical and world knowledge, which proved to be the most difficult contradictions to detect on a broad scale. While we are able to find certain specific relationships in the development sets, these features attained only limited coverage. Many contradictions in this category require multiple inferences and remain beyond our capabilities:

T: The Auburn High School Athletic Hall of Fame recently introduced its Class of 2005, which includes ten members.

H: The Auburn High School Athletic Hall of Fame has ten members.

Of the types of contradictions in category (2), we are best at addressing those formed via structural differences and factive/modal constructions as shown in Table 4.25. However, creating features with sufficient precision is an issue for these types of contradictions. Intuitively, two sentences that have aligned verbs with the same subject and different objects (or vice versa) are contradictory. This indeed indicates a contradiction 55% of the time on our development sets, but this is not high enough precision given the rarity of contradictions.

Another type of contradiction where precision falters is numeric mismatch. We obtain high recall for this type (Table 4.25), as it is relatively simple to determine if two

numbers are compatible, but high precision is difficult to achieve due to differences in what numbers may mean. Consider:

T: Nike Inc. said that its profit grew 32 percent, as the company posted broad gains in sales and orders.

H: Nike said orders for footwear totaled \$4.9 billion, including a 12 percent increase in U.S. orders.

Our system detects a mismatch between 32 percent and 12 percent, ignoring the fact that one refers to profit and the other to orders. Accounting for context requires extensive text comprehension; it is not enough to simply look at whether the two numbers are headed by similar words (grew and increase). This emphasizes the fact that mismatching information is not sufficient to indicate contradiction.

We handle single word antonymy with high precision (78.9%), as well as, negation. Nevertheless, Harabagiu *et al.* (2006)'s performance on detecting contradictions arising from negation and antonyms (64% on a balanced dataset) demonstrates that further improvement on these types is possible; indeed, they use more sophisticated techniques to extract oppositional terms and detect polarity differences. Thus, detecting category (1) contradictions is feasible with current systems. Since more than half of the examples found in the real corpus are of that category (60% if we restrict the corpus to the pairs coming from the GALE data), it suggests that we may be able to gain sufficient traction on contradiction detection for real world applications. Even so, category (2) contradictions must be targeted to detect many of the most interesting examples and to solve the entire problem of contradiction detection. Some types of these contradictions, such as lexical and world knowledge, are currently beyond our grasp, but we have demonstrated that progress may be made on the structure and factive/modal types.

Despite being rare, contradiction detection is foundational for information analysis. Our detailed investigation demonstrates which aspects of it can currently be resolved and where further research must be directed.

Acknowledgements

We would like to thank Sebastian Padó for all his work as part of the Stanford Distillation team.

4.5 Distillation in an Integrated GALE system

4.5.1. Using Query-Time Information to Improve Multi-lingual Search and Response Generation

Authors: Kristen Parton, Weiyun Ma, Kathleen McKeown, and James Allan

4.5.1.1. Introduction

Cross-Lingual Question Answering (CLQA) typically focuses on the task of retrieving answers in language **f** (or a set of languages **F**) that are relevant to a question

posed in language *e*. The GALE distillation task expands this task to include result translation - we call this new task *translingual* question answering. In many real life scenarios, such as intelligence work, international business, or tourism, users may not be able to read language *f* and therefore need results in the query language, *e*. Whereas, result translation in CLQA has usually been considered a separate, post-processing step that should be evaluated separately, in translingual QA, relevance and translation quality are inseparable: a poor translation of a relevant answer is likely to be deemed irrelevant.

To address this problem, we propose a post-editing mechanism to deal with translation errors in answers. We first use a machine translation (MT) system to translate a source sentence, and then detect and correct problematic translations using information available in the QA context. In this framework, we treat MT as a black-box, and focus on improving the output rather than re-engineering the translation system itself. As MT is used in a more and more multi-lingual applications, this situation will be quite common.

4.5.1.2. Post-Editing Framework

Human post-editing is commonly used in commercial applications of MT, to correct errors in MT output. Knight and Chander (1994) introduced the idea of an automatic post-editing system that would be “portable across MT systems.” More recently, Simard *et al.* (2007) use a statistical machine translation (SMT) system to learn to post-edit the output of a rule-based MT system. Elming (2006) applied transformation-based learning to learn substitution rules for post-editing an MT system. Similar to that work, we focus on two kinds of MT errors: named entity mistranslation and verb deletion. This post-editing framework is applicable across a variety of MT systems, since it only needs word alignments.

Within the context of the GALE Distillation task, our system works as follows. The query is sent to the information retrieval system for document retrieval. Each document retrieved is returned in both the source language and the query language (via MT). Based on the input query, the post-editing modules check each sentence for translation errors. When an error is detected, the translation is corrected (if possible). The corrected translations are then sent to response generators, which are responsible for selecting relevant sentences and generating a system response.

4.5.1.3. Post-Editing Named Entities

Named entities are particularly important in the GALE distillation task: in the second year GALE evaluation, more than 90% of the arguments contained named entities. However, SMT systems typically perform poorly on named entity translation, as many named entities are out-of-vocabulary with respect to the training data. Since the GALE corpus contains informal genres, there are many variations of the same name, such as misspellings and slang, making it even harder for MT systems to correctly translate the names. Incorrect name translation adversely affects both the precision and recall of GALE distillation. The recall is decreased because the QA component cannot find as many answers containing the named entity in the query. The precision is decreased when the translated answers have incorrect named entity translations, and therefore, appear irrelevant.

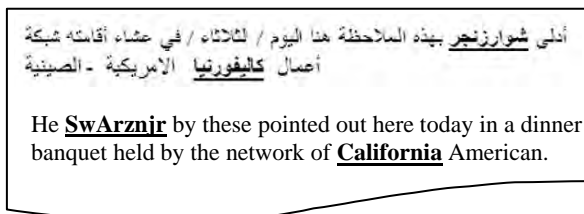
Our approach addresses both of these problems. We improve our retrieval by simultaneously searching for a query in the source documents (in Chinese and Arabic), as well as, in their translations (in English). The query translation is based on dictionary lookup, but the dictionary we use is derived from Wikipedia, which allows us to cover many different name variations. Ferrandez *et al.* (2007) demonstrate that Wikipedia is an excellent source of named entity translations for cross-lingual question answering, since it is created and edited by humans. The simultaneous multilingual IR (SMLIR) system that we use is described more fully in Parton *et al.* (2008).

When sentences are returned that contain a name in the source language, but not in translation, we hypothesize that the name was mistranslated. Using word alignments, our system replaces the incorrect name translation with the name from the original query.

There are several reasons that SMT post-editing could improve over SMT. First, statistical machine translation takes a sentence-by-sentence approach to translation, ignoring issues of coherence and consistency. In one document in our corpus, the same name was translated three different ways. A name that co-referred in the source language appeared to be three different people in the translation. Second, we have more information at query-time than at document translation time. Since we run named entity tagging on the queries, we can try to match them to documents that are name-tagged in both languages. Therefore, we are using information from multiple sources to make an informed translation decision. This is similar to the approach of Ji and Grishman (2007) for using joint inference over information extraction and entity translation to improve name translation.

4.5.1.3.1. Algorithm

Consider a query such as “Provide information on [Arnold Schwarzenegger]”. Using our Wikipedia-derived dictionary, we can find English variations of this name (e.g., slang variants such as “the governor” and misspellings like “schwartzenegger”), as well as, Chinese and Arabic translations of the name and name variations. Using the set of all English, Chinese and Arabic terms for “Arnold Schwarzenegger”, we issue a SMLIR query to simultaneously search the corpus and the translated corpus. An example result is shown in Figure 4.23. This sentence is relevant in Arabic, but a user is unlikely to rate it relevant in English given the poor translation.



Actual translation: “Schwarzenegger made this statement here today, Tuesday, at the dinner held by the Chinese-American business association of California.”

Figure 4.23: A translated answer for the query [Schwarzenegger], with Arabic source and English machine translation. A reference translation is shown below. Note that the word “Chinese” is deleted in the machine translation, and “Schwarzenegger” is mistranslated.

Using the SMLIR approach, we can detect potentially incorrect translations in a document when a document-language match is found for a query, but a query-language match is not found. In the example from Figure 4.23 the query translation matched the query “*Schwarzenegger*” in Arabic شوارزجر , but the document translation did not match the query due to a translation error. This occurs since the query translation and document translations are done by different methods. Since we have word alignments from the document translation system, after detecting an incorrect translation, we can replace the incorrect translation (“\$warznrj”) with the original query (“Schwarzenegger”). The user then gets a response with the correctly spelled name, and the translated document is perceived relevant.

Figure 4.24 illustrates the post-editing process for a different query. Our algorithm for query-directed SMT post-editing is:

1. Use SMLIR system to detect potential mistranslation. If a result contains a match in the foreign source but not in the English, consider it a potential error.
2. Using word alignments, extract the MT hypothesis: the English words that correspond to the foreign source match.
3. If the MT hypothesis matches a name variation in our dictionary, do not rewrite this translation.
4. Use word alignments to decide which translation tokens to replace. Name translations are not necessarily contiguous, so it is important not to insert the name multiple times. If the match is part of a larger phrase match, there may be links to other words, which should not be rewritten.

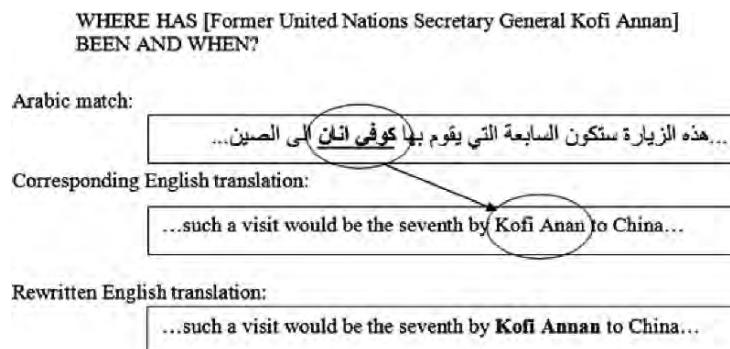


Figure 4.24: Query-directed statistical machine translation post-editing. SMLIR returned a match on the Arabic part of the document for “Kofi Annan”, but not on the English side. We use word alignments from SMT to extract “Kofi Annan” from the English translation. Since “Kofi Annan” is not a known name variation for “Kofi Annan”, we rewrite the sentence with the term as it appeared in the query.

In general, a find-and-replace approach to machine translation is too simplistic and likely to be problematic. We restrict our post-editing to proper names, which are more amenable to rewriting than arbitrary words or phrases. The approach is generally language-independent, although for some languages, additional work may be needed to address morphology (for example, case endings in Russian). We implemented the post-editing module in both Chinese and Arabic, but we evaluated on Arabic only.

4.5.1.3.2. Results

The goal of our automatic machine translation post-editing system is to use query time information to improve the translation quality of returned results. Our preliminary results indicate that this approach is able to improve our MT output. We ran the post-editing system on 127 Arabic GALE queries, using the top 10 document results from our SMLIR system. Of those, 28 (22%) of the queries returned documents that required post-editing. For the queries that were post-edited, 15% of the IR name matches were rewritten. For each query, up to the first 5 post-edits were examined by a student of Arabic. The annotator decided whether the replacement was Acceptable, Not Acceptable or Ambiguous. Of the 101 rewrites examined, our replacements were Acceptable 93% of the time. 6% were Not Acceptable and 1% was Ambiguous. Our post-editing algorithm was especially conservative, so it aimed for precision rather than recall. A more in-depth evaluation is required to explore this issue further.

Improved name translation is essential for good translingual applications like GALE Distillation. However, MT metrics such as the BLEU score (Papineni *et al.* 2002) or METEOR (Banerjee and Lavie, 2005) do not take into account the relative importance of various words in the sentence. Producing an incorrect translation of a name such as “Zarqawi” has the same effect on BLEU score as producing an incorrect determiner (“a” instead of “the”). On the other hand, a deleted determiner is unlikely to affect a user’s understanding of a GALE answer, whereas a mistranslated name can make a relevant sentence seem irrelevant to a user. By using query-directed post-editing, we have shown that we can improve named entity translation for translingual applications.

4.5.1.4. Post-Editing Deleted Verbs

Our second post-editing module is aimed at detecting and correcting verb deletion. The main verb usually plays a crucial role in conveying the whole meaning of a sentence. A missing main verb can result in an incomprehensible sentence as seen in the first example of Figure 4.25 where the Chinese verb “被捕” was not translated at all. In other cases, a deleted main verb can result in miscommunication; in the second example of Figure 4.25, the Chinese verb “减退” should have been translated as “decreased”. An English native speaker could easily misunderstand the meaning to be “People love classical music every year.” which is the opposite of the original intended meaning.

In cases where only the main verb is missing, MT evaluation metrics (such as BLEU or Translation Edit Rate) may give a high score, but the sentence can nonetheless be incomprehensible. We propose a novel method which uses redundant information available within QA to detect and correct verbs deleted in MT.

MT	On December 13 Saddam .
Ref	On December 13 Saddam was arrested .
Source	12月 13日 萨达姆被捕。
MT	People of classical music loving every year.
Ref	People's love for classical music decreased every year.
Source	民众对古典音乐的热爱逐年 减退 。

Figure 4.25: Examples of verb deletion in Chinese-English SMT

One of the recent trends in phrase-based SMT is to integrate more syntactic considerations into translation models. SMT systems integrate syntactic information of a source side (Watanabe *et al.* 2003; Wang *et al.* 2007) or of a target side (Yamada and Knight 2001; Charniak and Knight 2003) or of both sides (Huang *et al.* 2006; Lavie 2008) into their translation model. Most systems utilize *N*-grams as the language model for providing local constraints, whereas, global constraints are needed to check for a missing main verb in a target sentence. Our work tries to complement this kind of global consideration through post-editing mechanism using question-answering context.

We modify the set of related MT documents retrieved by IR so that the QA system can utilize these modified documents to generate answers. Handling this problem at query time rather than during SMT gives us valuable information which was not available during SMT, namely, a set of related sentences and their translations which may contain the missing verb. By using translation examples of verb phrases and alignment information in the related documents, we are able to find an appropriate English verb and embed it in the right position as the main verb in order to improve MT quality and QA performance. More details are described by Ma and McKeown (2009).

4.5.1.4.1. Algorithm

The post-editing algorithm works as follows: we detect a clause with a missing main verb, determine which Chinese verb phrase should have been translated, and find an example sentence in the related documents, which can be used to modify the sentence in question.

In the first step, we identify the main clauses, subordinate clauses and conjunct clauses from a translated English sentence. If a clause does not have any POS tag that can serve as a main verb (VB, VBD, VBP, VBZ), it is marked as missing a main verb. MT alignment information is then used to further ensure that these marked clauses are really missing main verbs. We segment and tag the Chinese source sentence using the Stanford Chinese segmenter and the CRF Chinese POS tagger developed by Purdue University. If we find a verb phrase in the Chinese source sentence that was not aligned with any English words in the SMT alignment tables, then we label it as a verb translation gap (VTG).

We translate VTGs by using verb redundancy in related documents: if the VTG was translated in other places in related documents, the existing translations can be reused. Related documents are likely to use a good translation for a specific VTG as it is used in a similar context. A verb's aspect and tense can be directly determined by referencing the

corresponding MT examples and their contexts. If a given VTG did not have any other translation record, then the VTG will not be processed.

Given a set of related MT documents and their MT alignments to a certain query, we collect all Chinese verb phrases and their translations along with their frequencies and contexts.

One key issue is to decide appropriate contextual features of a verb. A number of researchers (Cabezas and Resnik 2005; Carpuat and Wu 2007) provide abundant evidence that rich context features are useful in MT tasks. Carpuat and Wu (2007) tried to integrate a Phrase Sense Disambiguation (PSD) model into their Chinese-English SMT system and they found that the POS tag preceding a given phrase, the POS tag following the phrase and bag-of-words are the three most useful features. Following their approach, we use the word preceding and the word following a verb as the context features. We select the proper translation of a given VTG according to the Naive Bayes equation shown below

$$t' = \underset{t_k}{\operatorname{argmax}} P(t_k | pw, fw) \\ = \underset{t_k}{\operatorname{argmax}} (\log P(t_k) + \log P(pw | t_k) + \log P(fw | t_k))$$

where pw , fw and t_k represent the preceding source word, the following source word and a translation candidate of a VTG, respectively.

Finally, once we have selected an English translation, we use word alignment information to insert the translation back into the translated sentence. Chinese ordering differs from English mainly in the clause ordering. But within a clause centered by a verb, Chinese mostly uses a SVO or SV structure, just like English (Yamada and Knight 2001). Therefore, we assume the local alignment centered by a verb between Chinese and English is a linear mapping relation.

4.5.1.4.2. Results

Our test data is drawn from Chinese-English MT results generated by Aachen RWTH's translation system (Mauser *et al.* 2007), which is a state-of-art phrase-based SMT system. We passed five GALE queries to the IR module, and then post-edited all retrieved 18,886 sentences. Among these MT sentences, 1,142 sentences were detected and modified (6 % of all retrieved MT sentences).

Since we did not have reference translations of our corpus, we used human judgments to evaluate our post-editing module. We also believe that human judgments are more accurate than MT scoring techniques such as TER or BLEU, since readers can better tell whether the correct meaning was captured in translation. We recruited 13 Chinese native speakers who are also proficient in English to participate in the evaluation. Evaluators were presented with the Chinese sentence, the machine translation, and the post-edited machine translation in random order, so they could not tell which was post-edited. They were then asked to choose which translation was better, or "no difference". Each sentence was evaluated by three annotators.

We used majority voting (two out of three) to decide the final evaluation of a sentence judged by three people. Of the 1,142 modified sentences, 900 (79%) were judged better

than the original sentences. This is equivalent to 5% of all the retrieved sentences. For 629 (70%) of these 900 better modified sentences, all three evaluators agreed that the modified sentence was better.

We also evaluated the impact of post-editing on the 234 sentences returned by our response generator. Response sentences were judged as “Relevant”, “Partially Relevant”, “Irrelevant” and “Too little information to judge” by English native speakers. With our post-editing technique, 7% of 141 “Irrelevant” or “Too little information” responses became “Relevant” or “Partially Relevant.” None of the “Relevant” or “Partially Relevant” responses became “Irrelevant” or “Too little information” responses. Two examples of beneficial post-edits are shown below.

Question: Who/What is involved in Saddam Hussein's trial?

Original QA answer: On May 8, Saddam defense counsel in Jordan's capital Amman.

Modified QA answer: On May 8, Saddam defense counsel in Jordan's capital Amman *formally established*.

Original QA answer: According to the Iraqi transitional government announced that Saddam would on October 19.

Modified QA answer: According to the Iraqi transitional government announced that Saddam would on October 19 *to be tried*.

In total, the post-editing system improved 4% of the actual system response sentences, thus demonstrating that our correction of MT truly improves QA performance.

4.5.1.5. Conclusions

The GALE task gives us a unique opportunity to explore a *translingual* application, where doing well requires a correct answer and a readable translation of the answer. Our section builds on previous work in automatic post-editing of machine translation to correct errors in MT, with the goal of returning more readable, relevant answers. Like previous work, our approach treats MT as a black box, and does not rely on system-specific internals, so it is widely applicable to any MT system. Our work is novel in that it uses information available at query time to guide the post-editing process, thus tailoring the post-editing to the specific user needs. Our results showed that we can successfully detect and correct mistranslated named entities in a largely language-independent approach. We also demonstrated that we could detect verbs that were deleted by MT and insert correctly translated verbs back into the sentences. In the future, we plan to extend our approach to tackle other MT error types by using information available at query time.

Acknowledgements

We would like to thank Bob Armstrong for making the annotation happen, Mark Smucker and Giridhar Kumaran for help with INDRI and corpus issues. We would also like to thank Ben Carterette for help with discussions about evaluation. We would also like to thank the members of the NIGHTINGALE machine translation team for translation data, especially Nizar Habash.

4.5.2. Handling Out-of-Vocabulary Terms Using Query Driven Retranscription

Authors: Brian Kingsbury and Upendra V. Chaudhari

4.5.2.1. Introduction

Dealing with queries containing out-of-vocabulary (OOV) terms is a central problem in information retrieval from speech. Most speech transcription systems rely on closed recognition vocabularies that are small in comparison to the set of words that are encountered in operation. The OOV problem is especially severe for named entities, which are frequently the most important tokens to get right in information retrieval tasks. Moreover, OOV names often arise and become significant quickly and unpredictably. We first encountered this problem in the GALE Distillation task, where the goal is to provide concise, relevant responses to structured queries against a corpus derived from audio and text sources in Arabic, Chinese and English. In a Distillation evaluation, one of the test queries was:

FIND STATEMENTS MADE BY OR ATTRIBUTED TO [Isabelle Dinoire] ON [her successful face transplant].

Due to training data restrictions, only audio sources in the distillation corpus contained information on Isabelle Dinoire’s face transplant. However, the transcription system used to convert the audio to text did not have “Dinoire” in its vocabulary, so the Distillation System was unable to return any information in response to this query.

Phonetic indexing provides a partial solution to the OOV problem in that it can return a ranked list of segments that may contain the OOV terms; however, in the GALE Distillation task a list of possible hits is insufficient. What is required is text that can be further processed to provide the user with a concise and relevant answer to her query. This goal can be accomplished by augmenting the recognition vocabulary and language model of a transcription system with the OOV terms, and then decoding audio segments returned by a phonetic indexing system with the augmented recognizer. We refer to this process as query-driven retranscription.

4.5.2.2. System Architecture

Our system for query-driven re-transcription operates in two phases: indexing and retrieval. Both phases are illustrated in Figure 4.26. The indexing phase begins with transcription of the input speech. The transcription system, which is described in more detail below, produces three outputs: a diarization of the audio that includes a segmentation of the audio into shorter units and a clustering of segments into hypothesized speakers, a set of adaptation parameters for each speaker hypothesized in the diarization, and a one best word transcription of each segment in the diarization. The index is constructed by expanding the word-level transcripts into phone-level transcripts using the baseforms in the recognition dictionary. Lattice based indexing could be used, however in practice we have found that approximate search over one best transcripts is

more efficient and has comparable accuracy. Non-speech phones are discarded, then the phone-level transcripts are parsed into N -grams and an inverted index is constructed. For each N -gram that occurs in the audio corpus, the index stores all locations at which it occurs, where a location is a (segment, time) pair.

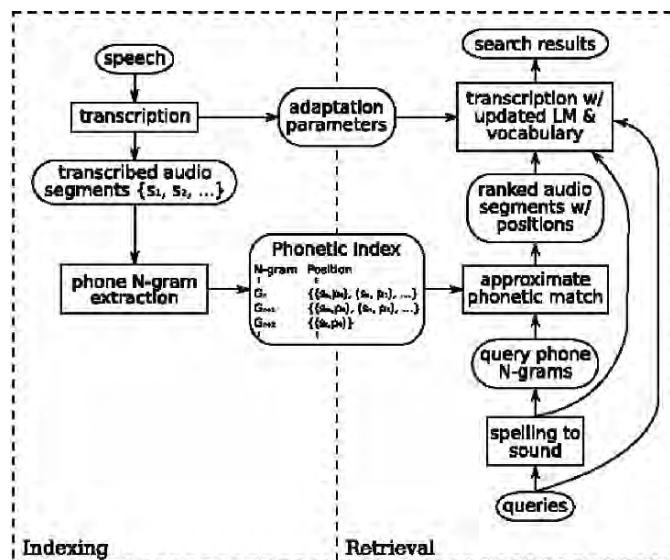


Figure 4.26: Architecture of the query-driven retranscription system.

For retrieval, query terms are converted into phone sequences. In-vocabulary terms are converted by looking up baseforms in the recognition dictionary, while out-of-vocabulary terms are converted using an automatic spelling-to-sound conversion system (Chen 2003).

The phonetic search, which is described in more detail below, is based on an approximate matching score that accounts for confusions between phone unigrams, bigrams and trigrams (Chaudhari and Picheny 2007). The match score for a segment is the average of the scores of the best-scoring index element for that segment for each n -gram in the query. The output of the phonetic search is a ranked list of segments that may contain the query term.

The baseforms from the spelling-to-sound conversion are added to the recognition lexicon and the OOV terms are added to the recognition language model. In the tests described below, the OOV terms are added to the language model as unigrams with the unknown word score. Finally, the audio segments that are identified by the phonetic search as possibly containing the query terms are retranscribed with the augmented system. The retranscription can be done efficiently because we need to re-decode only those segments that are identified by the phonetic search, and because we reuse the adaptation parameters computed during indexing and need to run only the final, speaker-adapted decoding pass.

4.5.2.2.1. Transcription System

The transcription system used in our tests operates in four stages: audio diarization, speaker-independent decoding, adaptation, and speaker-adapted decoding. The audio diarization stage takes as input a list of audio files and produces as output a list of segments with hypothesized speaker labels. The speaker-independent decoding produces a one best transcript of each segment in the diarization output. The adaptation stage uses the one best transcripts to compute a vocal tract length normalization (VTLN) warping factor, feature space maximum likelihood linear regression (FMLLR) transform, and up to 16 maximum likelihood linear regression (MLLR) transforms (mean only) for each speaker. The speaker-adapted decoding then produces a one best transcript of each segment, which is further processed by the audio indexing component.

The audio diarization process is similar to that described by Soltau *et al.* (2007), with two differences. First, before speaker clustering, we perform speaker change point detection using the modified Bayesian Information Criterion (BIC) method (Ajmera and Wooters 2003), over a four second sliding window. Second, hypothesized speech segments are padded by up to 0.5 seconds at the beginning and end of each segment. Due to constraints enforced throughout the audio diarization process, the resulting audio segments, which are treated as documents by the audio indexing system, are no more than 20.0 seconds long.

The raw acoustic features for transcription are 19-dimensional perceptual linear prediction (PLP) features. The speaker-independent pass mean-normalizes the PLP coefficients on a segment-by-segment basis, while the speaker adapted pass uses mean-and-variance-normalization on a speaker-by-speaker basis. Both recognition passes use linear discriminant analysis followed by a maximum likelihood linear transform (LDA+MLLT) to project from 9 spliced frames of raw features to a 40-dimensional recognition feature space. Both models use roughly 6000 quinphone context-dependent states. The speaker independent model uses 250K Gaussians, and the speaker-adapted model uses 300K. Both transcription systems are discriminatively trained using feature space minimum phone error (fMPE) (Povey *et al.* 2005) and minimum phone error (MPE) training. The acoustic models were trained on 430 hours of audio. We used lightly supervised training (Chan and Woodland 2004) on the TDT4 data.

The initial transcription of the corpus is done using statically compiled decoding graphs, but the retranscription is done using a dynamic decoder, because it would be prohibitively expensive to compile a new decoding graph for each query term. The dynamic decoder is similar to the one described by Soltau *et al.* (2001), but uses a statically compiled and minimized word network that allows for contexts spanning more than one word and allows for multiple pronunciations. We used two language models: a small language model (3.3M n -grams) for constructing static decoding graphs, and a large language model (54M n -grams) for in-memory lattice rescoring during the speaker-adapted transcription pass and for dynamic decoding during retranscription. Both language models are interpolated back-off four gram models smoothed using modified Kneser-Ney smoothing. The language models were trained on a collection of 335M words. The recognition lexicon comprises 84K words and 90K pronunciation variants.

This transcription system has a word error rate of 13.2% on the RT-04 evaluation set. While this error rate is slightly higher than the best published results on the RT-04 set (Gales *et al.* 2006), this level of transcription accuracy is achieved with an extremely simple transcription system that performs only two decoding passes and does no system combination.

4.5.2.2.2. Phonetic Indexing

The input to the query-driven retranscription is a ranked list of hypothesized term occurrences obtained via an approximate search over an index built on phonetic. Ranking is based on the match score, defined below.

Building an Index

The indexing phase takes as input the one best word transcripts produced by the transcription system. These are converted into phone transcripts using the pronunciation dictionary. The diarization consists of segments that are, on average, about 15 to 20 seconds long. In the indexing scheme, each segment is considered to be a unique document, D_k , whose length is measured in phones. These are the units that are reported on the initial ranked list of possible term occurrences. Using the set of documents defined above, an inverted index is created with a temporal granularity corresponding to the positions of phones. The phone sequences in the index are five phones long ($N = 5$). Each document D_k is parsed, using a sliding window, into a sequence of n -grams that occur at each position:

$$D_k \rightarrow \{h_{k,1}, h_{k,2}, h_{k,3} \dots\},$$

where $h_{k,j} = \{p_{k,j}, p_{k,j+1}, \dots, p_{k,j+4}\}$ and each $p_{k,j}$ is an element of the phonetic alphabet. At the end, where full n -grams are not available, sub- n -grams are used.

$$I = \bigcup_k D_k$$

is the set of all unique n -grams that occur. The inverted index associates to each element of I the list of (document, position) pairs at which that element occurs. Note that $\mathbf{h} \in I$ represents an observed N -gram and is treated as a hypothesis vector. The query is considered to be the reference: the exact sequence of phones that is sought.

Confusion Matrix

Phone confusions are represented as $P(p_i|p_j)$, the probability that p_i is the true phone when p_j is observed. Our approach also incorporates high order confusions, given by $P(p_i p_j p_k | p_l p_m p_n)$ or $P(p_i p_j | p_l p_m)$, the probability of trigram or bigram confusions. The distributions $P(p_i \dots | p_j \dots)$ are estimated from held-out data.

Generating Scores

The query is an arbitrary sequence of words. To perform search, each word in the query is expanded into one or more phone sequences via lookup in the pronunciation lexicon and via automatic baseform generation using a joint maximum entropy model trained on the Pronlex lexicon and Random House dictionary (Chen 2003). This allows us to handle out-of-vocabulary terms and obtain alternate pronunciations for query terms. Let Q_k denote a single phone sequence for a query term.

$$Q_k \rightarrow \{\mathbf{r}_{k,1}, \mathbf{r}_{k,2}, \mathbf{r}_{k,3} \dots\},$$

where $\mathbf{r}_{k;j} = \{p_{k,j}, p_{k,j+1}, \dots, p_{k,j+4}\}$ is the phone 5-gram at position j in Q_k . Given \mathbf{r}_i and $\mathbf{h}_j \in I$ a match score $S(\mathbf{r}_i, \mathbf{h}_j)$ is computed (Chaudhari and Picheny, 2007). Then, given a document D and a query n -gram \mathbf{r} , the best matching index element corresponding to the document is given by

$$\mathbf{h}^*(D, \mathbf{r}) = \underset{\mathbf{h} \in I/D}{\operatorname{argmax}} S(\mathbf{r}, \mathbf{h}),$$

Letting N_Q = the number of n -grams extracted from the query, the score is

$$\frac{1}{N_Q} \sum_{\mathbf{r}_i \in Q} S(\mathbf{r}_i, \mathbf{h}^*(D, \mathbf{r}_i))$$

4.5.2.3. Experiments

To evaluate the effectiveness of query-driven re-transcription in retrieving OOV terms, we performed a set of experiments on the English audio data used in the GALE Distillation evaluation. This collection of 25 hours of audio comprises 53 broadcasts on CNN, MSNBC, and NBC from the period of January 3, 2005 until June 30, 2005. Initial transcripts of all 53 broadcasts, excluding advertisements, were produced by a single transcriber. In a quality-control phase, a second transcriber examined all occurrences of words that were not present in the recognition vocabulary of the transcription system. Where these OOV words corresponded to transcription errors, the errors were corrected. In cases where a genuine OOV word was found, it was verified and, in the case of proper nouns, checked against online resources such as Wikipedia to validate spelling. Excluding disfluencies and partial words, we found 845 occurrences (tokens) of 430 OOV words (types) in the 200411 words of the transcripts. 70% of the types and 82% of the tokens derived from names. The transcription word error rate on this data set is 15.0%.

We treat each individual OOV word as an independent query, and measure the retrieval performance of the re-transcription system on the entire set of queries in terms of precision, recall, F_1 -measure, and mean average precision (MAP). Scoring is done at the level of speech segments, as defined by the audio diarization component of the transcription system. This experimental design underestimates the performance of re-transcription in an actual application setting where the queries are likely to span multiple words, because both phonetic indexing and the end-to-end re-transcription system perform better for longer queries.

For each query, the spelling-to-sound system produces up to five pronunciation variants, with most words receiving only one. On average, there are 1.75 variants per word. Each pronunciation variant is treated as a separate query by the audio indexing system, which produces a ranked list of 100 candidate locations for each query, as described in Section 4.5.2.2.2. When a query word has more than one variant, the lists of potential hits from the different variants are merged: because an approximate phonetic match is used, different variants can match the same audio segment. Next, for each query, the potential hits are retranscribed as described in Section 4.5.2.2. If the output of the retranscription contains the query term, it is retained; otherwise, it is discarded.

Metric	audio indexing	Retranscription
Precision	0.0061	0.3985
Recall	0.4563	0.4113
F_1	0.0121	0.4048
MAP	0.2656	0.4137

Table 2.26: OOV term retrieval performance for the audio indexing and the full re-transcription system. Note that the audio indexing was turned for high recall to ensure good end-to-end system performance.

query length	count	precision	recall	F_1
≤ 3	35	0.3913	0.2571	0.3103
4	117	0.3101	0.3419	0.3252
5	210	0.3212	0.2952	0.3077
6	185	0.3578	0.3946	0.3753
7	120	0.4274	0.4167	0.4219
8	92	0.3740	0.5326	0.4395
9	49	0.6047	0.5306	0.5652
≥ 10	37	0.8857	0.8378	0.8611

Table 4.27: OOV term retrieval performance for the query-driven re-transcription system, as a function of query length in phones.

The performance of the system, both after audio indexing and after re-transcription, is summarized in Table 2.26. It is important to note that none of the target query terms could have been returned by the original transcription system because all the targets were OOV, and also that the audio indexing was tuned for high recall and low precision because it was followed by re-transcription-based filtering. Table 4.27 shows that retrieval performance improves as the length of the query increases. This pattern is not surprising because longer queries are more distinctive.

4.5.2.4. Conclusions and Discussion

We have shown for English broadcast material that it is possible to combine audio indexing and speech transcription to identify audio segments containing terms that are out of the transcription system's original vocabulary and provide transcripts for those segments that contain the target OOV terms. While the accuracy with which these terms are transcribed is modest ($F_1 = 0.40$), this is a substantial improvement over the baseline case in which none of the terms would appear in the speech transcripts. Also, in actual use we expect the query terms to be longer than the average (single word) queries in this

study, which should lead to better retrieval performance, as shown in Table 4.27, there are still many other issues that need to be addressed, however, for query-driven retranscription to be a useful addition to practical Distillation Systems.

Is the text around out-of-vocabulary terms sufficiently well-formed that it is usable by distillation engines? Producing the OOV query terms in the correct speech segments is only the first step in handling OOV queries against audio data in a Distillation System. The text surrounding the OOV query must be sufficiently accurate and well-formed that the modules in a Distillation System can correctly process it. The current study does not address this question. It is quite likely that the addition of the OOV terms to the transcription system's language model as unigrams having the same score as the unknown word will not provide enough prior information to ensure accurate transcription. We believe that a better approach would be to define classes for various entities that might show up as out-of-vocabulary terms, e.g., personal names, organization names, and place names, and to assign new OOV terms to the correct class with a score appropriate for an unknown word in that class. This would model the OOV terms in context, and should improve both the retrieval of the OOV terms, as well as, the accuracy of the transcribed text surrounding the OOV terms.

How can retranscription be applied to translational distillation? There are additional challenges that must be overcome for query-driven retranscription to work in a translational setting. First, we need methods for producing sensible phonetic search queries from English-language terms. We believe that this problem could be solved by using statistical models that perform transliteration (in the common case of names) from English to the source language or that detect common co-occurrences of names (or word class elements) in parallel corpora. Note that such models would probably need to distinguish between terms that originate in the source language and are presumably transliterated into English queries, terms that originate in English, and terms that originate in some third language (e.g., searching for an Arabic personal name in Chinese news broadcasts). Second, we need methods to add new words to machine translation systems, much in the way that the new words are added to the transcription system. We think that simple approaches can work, especially if the Distillation Engine performs source-language processing.

How broadly can retranscription be applied in Distillation Systems? In principle, query-driven retranscription can be used any time the query contains terms that are not present in the audio transcripts or the translations of the audio transcripts, even in cases where the missing terms are in-vocabulary. If target terms are indeed present in the audio, but are absent from the input to the Distillation System, the cause may be some problem in the transcription or translation system such as incorrect pronunciations, poor language models, or poor translation models. The retranscription process, by using a larger set of possible pronunciations for a term and less restrictive language and translation models, may recover missing terms. The cost of this improvement in recall is reduced precision, and work is required to determine the best tradeoffs.

Can the information gained in retranscription be fed back into the upstream transcription and translation systems to improve their performance? Clearly it would be useful to use the information gained from user queries about important named entities and other query terms that are out of vocabulary to extend the vocabularies and

language models of the upstream transcription and translation systems. One option would be to simply leave the added terms in the transcription and translation models. This approach is unlikely to work. We did something similar to this in our experiments with the 2006 GALE data when we processed all of the queries in a single batch. That is, we added all of the pronunciations of all of the OOV terms to the transcription vocabulary and language model, and retranscribed all segments identified by the audio indexing as potential hits. Retrieval performance was much worse than when we processed each query individually: $F_1 = 0.30$, primarily due to a 50% reduction in precision. A more practical approach would be for a deployed system to track the OOV query terms, and for the lists of these terms to be used to extend the transcription and translation models when they are periodically updated.

Acknowledgements

Thanks to Radu Florian for alerting us to the Isabelle Dinoire problem in the 2006 evaluation and insisting that we find a solution.

4.5.3. Name Extraction and Translation for Distillation

Authors: Heng Ji, Ralph Grishman, Dayne Freitag, Matthias Blume, Zhiqiang (John) Wang, Shahram Khadivi, Richard Zens, and Hermann Ney

4.5.3.1. Introduction

Traditional MT systems focus on the overall fluency and accuracy of the translation, but fall short in their ability to translate certain informationally critical words. In particular, the translation of names is fundamentally different from the translation of other lexical items. Table 4.28 shows the wide range of cases that must be addressed in translating Chinese names into English, according to whether a name is rendered phonetically (P), semantically (S), or a mixture of both (M).

We may be expected to translate source language tokens which do not appear in the training corpus, based on our knowledge of transliteration correspondences (e.g., “*You shen ke*” to “*Yushchenko*”) and of contexts in the target language (e.g., to distinguish “*Yasser Arafat*” from “*Yasir Arafat*”). In addition, some source names may appear in abbreviated form and may be mistranslated unless they are recognized as abbreviations. For example, “以” is the abbreviation for ‘*Israel*’, but can also be translated into the common word ‘*as*’. Furthermore, errors may be compounded when part of an out-of-vocabulary name is mistakenly segmented into common words. For example, “*瓦斯涅夫斯基(Kwasniewski)*” is mistakenly translated into “*gas(瓦斯) Novsky(涅夫斯基)*” by a phrase-based statistical MT system; “*博贝列夫 (Bobylev)*” receives incorrect translations from different MT systems because it is not recognized as a name: “*German Gref*”, “*Bo, yakovlev*”, “*Addis Ababa*”, “*A. Kozyrev*” and “*1988 lev*”.

Name Translation Types		Chinese	English
Context/ Ethnic Independent	P → P	尤申科 (<i>You shen ke</i>)	<i>Yushchenko</i>
	P → S	可伶可俐 (<i>Ke Ling Ke Li</i>)	<i>Clean Clear</i>
	P → M	欧佩尔吧	<i>Opal Bar</i>
	S → P	旧金山 (<i>Old Golden Mountain</i>)	<i>San Francisco</i>
	S → S	解放之虎	<i>Liberation Tiger</i>
	S → M	长江(<i>Long River</i>)	<i>Yangze River</i>
	M → P	清华大学学报 (<i>The Journal of Tsinghua University</i>)	<i>Tsinghua Da Xue Xue Bao</i>
	M → S	百新百网站	<i>Best Buy Website</i>
	M → M	尤干斯可石油天然气公司	<i>Yuganskneftegaz Oil and Gas Co.</i>
Context/ Ethnic Dependent	P → P	亚西尔阿拉法特	<i>Yasser Arafat (PLO Chairman)</i>
			<i>Yasir Arafat (Cricketer)</i>
		潘基文	<i>Pan Jiwen (Chinese)</i>
	<i>Ban Ki-Moon (Korean Foreign Minister)</i>		
	S → S	红军	<i>Red Army (in China)</i>
			<i>Liverpool Football Club (in England)</i>
	M → M	圣地亚哥市	<i>Santiago City (in Chile)</i>
<i>San Diego City (in CA)</i>			

Table 4.28: Examples for Diverse Name Translation Types.

Name translation is important well beyond the relative frequency of names in a text: a correctly translated passage, but with the wrong name, may lose most of its value. Many GALE distillation templates involve names, so name processing is the key for accurate distillation from foreign languages. We found that distillation performed notably worse on machine translated texts than on texts originally written in English, and our error analysis indicated that a major cause was the low quality of name translation.

Thus, it appears that better entity name translation can substantially improve the utility of machine translation and the amount of information that can be gleaned from foreign sources. To meet these challenges, the Nightingale team has built a name translation component which operates in tandem with a conventional phrase-based statistical MT system, identifying names in the source text and proposing translations to the MT system. Versions have been developed for both Chinese-to-English and Arabic-to-English name translation. The Chinese system is more fully developed, so – while we will describe the name taggers for both languages – this section will focus on the translation of Chinese.

4.5.3.2. System Overview

The overall system pipeline for Chinese is summarized in Figure 4.27 (the system for Arabic has a subset of these components). This system runs a source language name

tagger, then uses a variety of strategies to translate the names it identifies. We shall present each of the main components in the following sections.

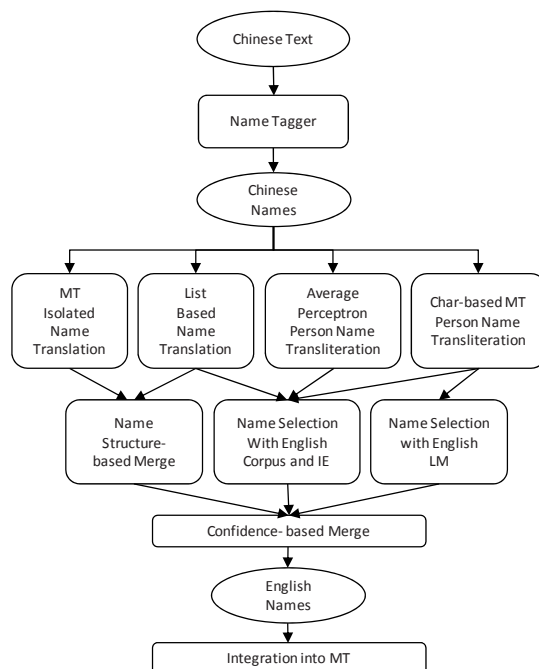


Figure 4.27: Chinese System Overview.

4.5.3.3. Name Tagging

Both Arabic and Chinese name taggers are trained on several years of ACE (Automatic Content Extraction) corpora, and can identify names and classify them as PER (persons), ORG (organizations), GPE (‘geo-political entities’ – locations which are also political units, such as countries, counties, and cities) and LOC (other locations).²⁰ In the following we will describe the Arabic and Chinese entity extraction systems.

4.5.3.3.1. Arabic Name Tagging

To identify Arabic names we trained a structured perceptron model, as detailed by Farber *et al.* (2008). Structured perceptrons are in a class of models, which also includes conditional random fields and hidden Markov Support Vector Machines that can exploit arbitrary features of the input in search of an optimal assignment of labels to a given input. We compensate for Arabic’s lack of name-relevant orthographic features (i.e., capitalization) and its relative lack of lexical resources (e.g., gazetteers) by expanding the feature set available to the model. First, we derive term clusters through a statistical analysis of a large volume of unlabeled Arabic newswire text, and treat a given term’s cluster membership as a feature (Freitag 2004). Second, we apply the MADA tool (Roth

²⁰ <http://www.nist.gov/speech/tests/ace/>

et al. 2008) for Arabic morphological analysis and word sense disambiguation. Two Boolean features are derived from the output of MADA, the first reflecting whether the analysis is successful (whether the word was in or out of vocabulary), and the second indicating whether the English gloss returned by MADA is capitalized.

4.5.3.3.2. Chinese Name Tagging

The Chinese name tagger consists of a Hidden Markov Model (HMM) tagger augmented with a set of post-processing rules (Ji and Grishman 2006). The HMM tagger generally follows the Nymble model (Bikel *et al.* 1997). Within each of the name class states, a statistical bigram model is employed, with the usual one-word-per-state emission. The various probabilities involve word co-occurrence, word features, and class probabilities. Since these probabilities are estimated based on observations seen in a corpus, several levels of “back-off models” are used to reflect the strength of support for a given statistic, including a back-off from words to word features, as for the Nymble System. To take advantage of Chinese names, we extend the model to include a larger number of states, 14 in total. The expanded HMM can handle name prefixes and suffixes, and has separate states for transliterated foreign names. Finally a set of post-processing heuristic rules are applied to correct some omissions and systematic errors.

4.5.3.4. Candidate Name Translations

We first apply the source language name taggers to identify named entities, and then generate a set of candidate translations for each name. We describe here the techniques we used for Chinese-to-English translation.

4.5.3.4.1. MT Isolated Name Translation

As a baseline we translate each source phrase referring to a named entity (a name ‘mention’) in isolation into English using our MT system. The only difference from full text translation is that, as subsentential units are translated, sentence boundaries are not assumed at the beginning and end of each input name mention.²¹

The RWTH Aachen Chinese-to-English machine translation system (Zens and Ney 2004, Zens *et al.* 2005) is used. It’s a statistical, phrase-based machine translation system which memorizes all phrasal translations that have been observed in the training corpus. The posterior probability is modeled directly using a weighted log-linear combination of various models: an n -gram language model, phrase translation models and word-based lexicon models, as well as, a lexicalized reordering model. The model scaling factors are tuned on a development set to maximize the translation quality (Och 2003). The bilingual training data consists of about 8 million Chinese-English sentence pairs with more than 200 million words in each language. The language model was trained on the English part

²¹ We tried an alternative approach in which mentions are translated in context and the mention translations are then extracted using word alignment information produced by the MT system, but it did not perform as well. The word alignments are indirectly derived from phrase alignment and can be quite noisy. As a result, noise in the form of words from the target language context is introduced into the mention translations. Manual evaluation on a small development set showed that isolated translation obtains (about 14%) better F-measure in translating names.

of the bilingual training data and additional monolingual data from the GigaWord corpus, about 650 million English words in total. The Chinese text is segmented into words using the tool provided by the Linguistic Data Consortium (LDC).

4.5.3.4.2. Name Pair Mining

We exploited a variety of approaches to automatically mine about 80,000 name pairs, as follows.

- *Extracting cross-lingual name titles from Wikipedia pages.* We run a web browser (Artiles *et al.* 2008) to extract titles from Chinese Wikipedia pages and their corresponding linked English pages (if the link exists). Then we apply heuristic rules based on Chinese name structure to detect name pairs; for example, foreign full names must include a dot separator, Chinese person names must include a last name from a closed set of 437 family names.
- *Tagging names in parallel corpora.* Within each sentence pair in a parallel corpus, we run the Chinese name tagger as described in Section 4.5.3.3.2 and the NYU English name tagger (Grishman *et al.* 2005). If the types of the name tags on both sides are identical, we extract the name pairs from this sentence. Then at the corpus-wide level, we count the frequency for each name pair, and only keep the name pairs that are frequent enough. Each member of the name pair then becomes the translation of the other member. The corpora used for this approach were all GALE MT training corpora and ACE 07 Entity Translation training corpora.
- *Using patterns for Web mining:* we constructed heuristic patterns such as “Chinese name (English name)” to extract NE pairs from web pages with mixed Chinese and English.

In addition, we exploited an LDC bilingual name dictionary (LDC2005T34) and a Japanese-English person name dictionary including 20126 Japanese names written in Chinese characters (Kurohashi *et al.* 1994). The Japanese-English name dictionary is essential, because the English translations of Japanese names preserve their Japanese pronunciation instead of Mandarin pinyin and therefore cannot be transliterated appropriately. For example, for the name “中田英寿” with Mandarin pinyin as “Zhong Tian Ying Shou”, the Japanese-English dictionary can provide two possible translations “Hidetoshi Nakata” and “Hidetoshi Nakada”.

Besides full string matching, we also parse name structures (e.g., first name and last name for persons; modifier and suffix word for organizations) and then match each name component separately. The matching results for each name component are then merged with the baseline translations obtained from MT isolated translation (shown as name structure based merging in Figure 4.27). We have also developed a name ethnicity identification component based on heuristic rules so that we can match non-Asian person names by pinyin to enhance the matching rate.

4.5.3.4.3. Statistical Name Transliteration

If a name can be translated using one of the lists described above, name transliteration is not required. Note that the writing systems of both Arabic and Chinese make transliteration nontrivial. Because non-native names in both languages are typically rendered phonetically, we require methods for “back-transliteration” in order to recover likely English renderings: Chinese renders foreign names by stringing together words that approximate a name’s constituent sounds (e.g., “Bu Lai Er” is the pinyin representation for “Blair”), while Arabic omits short vowels. Both languages lack and must approximate some English sounds.

We adopted a data-driven approach to address this problem, and trained two foreign-to-English character transliteration models to generate multiple transliteration hypotheses (multiple plausible English spellings of a name from Chinese or Arabic), using bilingual name lists assembled from several sources, as described above in Section 4.5.3.4.2. We pursued two quite different approaches.

In one, we applied state-of-the-art statistical machine translation techniques to the problem, to translate the character sequence of a foreign name to a sequence of letters which form an English name. We created a 6-gram character based language model (LM) trained from a large list of English names to rank the candidate transliterations. In this approach, no reordering model is used due to the monotonicity property of the task, and model scaling factors are tuned for maximum transliteration accuracy.

In the other, we trained a structured perceptron to emit character edit operations in response to a foreign string, thereby, generating a Romanized version. The two approaches achieve comparable accuracy. A detailed description and empirical comparison of these approaches are presented by Freitag and Khadivi (2007). Experiments showed that the combination of the two achieved 3.6% and 6% higher accuracy than each alone.

4.5.3.5. Name Translation Selection

In general, the output of the above approaches is a set of candidate English names. In order to ensure that the output is an acceptable English name, we select the best translation from a large number of candidates using Information Extraction (IE) results and Language Models (LM) built from large English corpora.

4.5.3.5.1. Name Selection with English Corpus and IE

To choose amongst these transliterations we first consult a large English corpus from a similar time period and its corresponding IE results, similar to the techniques described by Al-Onaizan and Knight (2002) and Kalmar and Blume (2007). We prefer those name spellings which appear in the corpus and whose time of appearance and global context overlaps the time and context of the document being translated. Possible contexts include co-occurring names, nominal phrases, and document topic labels.

We first process a large English corpus for names, their corresponding titles, the years in which they appear, and the frequencies under different categories, resulting in a large database of person entities in English. A source name’s time of appearance and the translation of any co-occurring titles are then compared to the entries in this database for

any candidate translations. For GPE name translation, the associated country information of the target name is used for comparison with the mined lists. Weights are optimized to combine the edit distance, temporal proximity and context similarity metrics. The candidate translation ranked highest according to this combined score is then selected. For example, in the following Chinese text:

… 据国际文传电讯社和伊塔塔斯社报道，<PER>格里戈里·帕斯科</PER>的律师<PER>詹利·雷兹尼克</PER> 向俄最高法院提出上诉。

The name transliteration components generate the following candidates for the name “詹利·雷兹尼克 (zhan li . lei zi ni ke)”:

24.11	amri	28.31	reznik
23.09	obry	26.40	rezek
22.57	Rezi	25.24	Linic
20.82	henri	23.95	Riziq
20.00	henry	23.25	ryshich
19.82	genri	22.66	lysenko
19.67	djari	22.58	ryzhenko
19.57	jafri	22.19	linnik

In a large English corpus we find the following sentence: “**Genri Reznik**, Goldovsky's **lawyer**, asked Russian supreme Court Chairman **Vyacheslav Lebedev**....”. By matching the Chinese entity extraction results (“律师 (*lawyer*)” referring to “詹利·雷兹尼克”) against English IE results (“*lawyer*” referring to “Genri Reznik”), we can select “Genri Reznik” as the correct translation. Without re-ranking with global context, the transliteration component would have produced “Amri Reznik”, an incorrect translation.

4.5.3.5.2. Name Selection with English Language Models (LM)

In addition, we have built a unigram word-based LM from a large English name list to penalize those transliteration hypotheses which are unlikely to be English names.

For example, for the name “保尔森” in the sentence “财政部长保尔森访问中国 (*Paulson, the Treasury Secretary, visited China*)”, the transliteration component produces the following top hypotheses: “*Bauerson*”, “*Paulsen*”, “*Paulson*” and “*Baulson*”. We assign a low score to “*Bauerson*” and “*Baulson*” because they don’t exist in the English unigram LM.

Each of the above translation and re-ranking steps produces a scaled confidence score; at the end we produce the final n-best name translations with token based weighted voting.

4.5.3.6. Integration into Machine Translation (MT)

The source text, annotated with name translations, is then passed to a statistical, phrase based MT system. To integrate name translation results into MT, two critical decisions have to be taken: the position of the name in the target sentence and the translation of the name. The position of the name is decided by the MT system, and the

translation of the name can be performed with or without its context. Therefore, we have two approaches: a simple transfer method, and an MT derived method.

4.5.3.6.1. Simple Transfer-Based Integration

The first method simply transfers the best translation of the source name to the target side based on word alignment. This approach ensures all name translations appear in the MT output, so it can principally benefit the Distillation task. However, it doesn't take into account word reordering or the words found in a name's context. For example, in the following sentence, the temporal phrase structure ("before...") was broken by the name translation transfer and therefore cannot be translated correctly:

Name Translation Annotated Chinese Sentence:

在("at")<NAME TYPE="PER" TRANSLATION="Zhao Ziyang">赵紫阳</NAME> 十四日陷入昏迷前("before), <NAME TYPE="PER" TRANSLATION="Tian Jiyun"> 田纪云</NAME>去看过他。

Reference Translation:

Before Zhao Ziyang fell into a coma on the 14th, Tian Jiyun visited him.

MT After Simple-Transfer based Name Integration:

At Zhao Ziyang before the 14th fell into a comma, Tian Jiyun went to visit him.

4.5.3.6.2. Phrase Table and LM based Integration

Therefore, we adopt the second method for improving full-text MT. The MT system considers the provided list of name-entity translations as a secondary phrase table, and then uses this phrase table, as well as, its own phrase table and language model to decide which translation to choose when encountering the names in the text. In order to avoid the problem of word segmentation inconsistency, we add all possible segmentations for each name. In order to obtain the best translation BLEU score, the name phrase table receives a coefficient optimized on a development set.

We then apply a 4-gram word based English LM to select the final translation from the name translation or MT phrase tables. For the example below, assuming name translation suggests a translation 'Paulson' while the regular MT phrase table produces 'Paulsen', then we can compare the LM scores based on the following phrases in the LM training corpus:

For "Paulson":

Paulson , Jr. to be the 74th Secretary of the Treasury

Paulson , the Treasury secretary , is a good guy

Treasury Secretary Henry M. Paulson, the former head of Goldman Sachs

Listen to US Secretary of the Treasury Henry M. Paulson discuss economic issues

For "Paulsen":

Paulsen was worried that her vacation in Los Angeles

Paulsen , chief investment officer at Norwest Investment Management

Paulsen , chief investment strategist at Wells Capital Management

Paulsen , professor of agronomy at Kansas State University

We then can choose "Paulson" as the best transliteration because its context "*Paulson, the Treasury*" matches those in the English LM.

4.5.3.7. Experimental Results and Analysis

In this section we present the overall performance of our Chinese-to-English name translation system.

4.5.3.7.1. Data and Scoring Metric

We conduct experiments on the text set of the NIST 2005 MT evaluation. By annotating names in the four reference translations, we identify on average 592 Person names, 1275 GPE names and 595 organization names - 2762 names in total. We apply the RWTH Aachen statistical phrase based MT system (Zens and Ney 2004; Zens *et al.* 2005) as our baseline, which uses the uniform translation model for names and non-names.

The following metric is defined to measure the accuracy of name translation:

$$\text{Approximate Accuracy} = \frac{(\# \text{ Target reference names found in MT Output})}{(\# \text{ Total reference names})}$$

Using a manually assembled name variant table, we also support the matching of name variants (e.g., “World Health Organization” and “WHO”).

4.5.3.7.2. Results

System Type	Y1 Baseline	Y2 Baseline	Y2 1Best-Phrase	Y2 Simple-Transfer	Y2.5 1Best-Phrase	Y2.5 NBest-Phrase	Y2.5 Simple-Transfer
PER	59.63	58.28	66.89	69.59	67.91	62.84	70.44
GPE	92.24	93.18	93.25	94.12	93.49	93.18	94.27
ORG	78.15	84.37	84.71	84.54	85.88	84.54	85.88
ALL	83.00	84.54	86.50	87.44	87.07	85.56	87.98

Table 4.29: Approximate Accuracy of Name Translation (%).

Table 4.29 summarizes the approximate accuracy results for Y1 and Y2 baselines, and the MT system integrated with name translation by two different approaches: simple transfer and name phrase table. Table 4.29 shows that the name translation system provided a 29.3% relative error reduction on overall name translation, and a 26.8% relative error reduction on person names using simple transfer integration. It’s also interesting to see that adding N-Best name translation does not provide improvement over 1-Best, which indicates that the name re-ranking approaches described in Section 4.5.3.5 are effective to select the best translations. Experiments also show that the 1-Best name phrase approaches can achieve about 0.2%-0.3% improvement on BLEU score over the MT baselines. In addition, we have found that although the Simple Transfer outperforms Phrase-table/LM based integration on approximate accuracy because it strictly preserves names, it achieved lower BLEU scores due to the cost of incorrect word ordering and loss

of function words. In the future we shall evaluate the impact of these effects on the Distillation task.

4.5.3.8. Impact on Cross-Lingual Sentence Retrieval

In this section we describe an experiment to measure the impact of name translation on cross-lingual spoken sentence retrieval: given a person name query in English the system should retrieve the sentences containing this name from Mandarin speech.

We used part of the GALE Y2 audio MT development corpus as our candidate sentence set (in total 668 sentences drawn from 45 shows). Fifty-three queries were constructed by selecting person names from the reference translations of reference transcripts for these shows. For each query, relevant sentences from the entire corpus were then manually labeled as answer keys, using reference translations as the basis for selection.

For each query, we then search the documents as translated by Machine Translation and by Name Translation. We compute the matching confidence between the query and a sentence substring based on the edit distance, with each operation assigned unit cost. We then define the sentence-level match confidence as the maximum confidence between the query and any sentence substring starting and ending on a token boundary.

We found that, over a range of confidence thresholds, searching name translation output yields comparable precision with an absolute improvement in recall of about 23% over searching MT output. More details are described by Ji *et al.* (2009).

4.5.3.9. Related Work

Most of the previous name translation work combined supervised transliteration approaches with Language Model based rescoring (e.g., Al-Onaizan and Knight 2002; Huang *et al.* 2004). Our approaches of mining name pairs from parallel corpora and mixed-code web texts are also related to the parenthesis translation mining as described by Lin *et al.* (2008). However, to the best of our knowledge, this is the first work that combines all of these existing techniques, as well as, name translation selection using IE techniques.

4.5.3.10. Discussion

The name extraction and translation system reduces the number of incorrectly translated names by about 30% (from 17% to 12%). People names remain the most difficult: even with name translation, the error rate remains about 30% (reduced from 40%). We analyzed the sources of these remaining errors for person names. They reveal the following major shortcomings of both name extraction and name translation.

4.5.3.10.1. Improving Name Extraction

Chinese name extraction errors contribute 5.5% to the person name translation error rate. As noted in Ji and Grishman (2006), boundary errors are dominant in Chinese name identification due to word segmentation errors. In addition, without indicative contexts,

the name tagger tends to miss some rare names and confuse foreign Person and GPE names. In order to address these problems we will exploit character based HMM and propagate multiple name tagging hypotheses to name translation to increase recall.

Automatic Speech Recognition (ASR) errors bring more challenges to name tagging; we will attempt to develop a phone-aware name tagger, using co-reference resolution to correct name ASR errors.

4.5.3.10.2. Improving Name Transliteration

In Ji and Grishman (2009) we demonstrated that feedback from name translation can be used to improve name tagging. We plan to extend the approach by incorporating more feedback from name transliteration confidence and mined name pairs as features.

Limitations in the name transliteration models contributed 8.8% to the translation error rate. It may be very challenging for our edit-distance based models to insert consonants at the end of syllables. For example, “Abdelrahman” is mistakenly transliterated into “Abderaman”. In the future we intend to use cross-lingual Wikipedia titles to capture more name pairs, especially more Arabic-origin and Russian-origin names.

4.5.3.10.3. Improving Name Translation Mining

A 5.8% error rate was due to the mismatch of some famous person names in uncommon spellings. For example, “鲍尔 (Powell)” does not exist in our mined name lists but its more common spelling “鲍威尔” appears with correct translation. Therefore it will be important to harvest name clusters by using within-document and crossdocument co-reference, so we that we can provide the clusters instead of each individual name to the name translation pipeline as input. In addition, we are also interested in automatically mining name translation pairs from comparable corpora as described by Sproat *et al.* (2006) and Klementiev and Roth (2006).

4.5.3.10.4. Improving Name Translation Selection

Our novel approach of using English IE to re-rank name translation results produces promising results (Kalmar and Blume 2007). However, it also relies heavily on the characteristics and size of selected English corpora to provide such ‘background knowledge’. We found that 4.6% in the error rate can be traced to a lack of corresponding English contexts for the name candidates. In the future, besides exploiting more unstructured corpora, we will also attempt to incorporate structured knowledge such as the social network databases existing on the web.

4.5.3.10.5. Toward Name Aware MT

In the experiments reported here, the underlying statistical MT system treats names just like any other tokens. Ultimately we will explore a new approach of training a name-aware MT system based on coordinated name annotation of source and target bitexts and conducting information driven decoding.

4.6 Evaluating Distillation Technology

4.6.1. Assessing Distillation Technology

Authors: Connie Fournelle, Olga Babko-Malaya, James V. White, and Dan Hunter

4.6.1.1. Introduction

While those tasked with collecting and analyzing military intelligence are often required to provide accurate, complete intelligence reports in a limited amount of time, they are typically constrained in this task by (1) language, which limits search to English-only sources of information, since translation is often not available in the requisite time frame, and (2) data volume, with extremely large data collections potentially masking important content by the sheer quantity of data available. To respond to a request for information, an analyst must find as much information as possible, as well as provide source traceability and confidence indicators, such as the existence of corroborating evidence.

GALE translation engines were designed to ingest available data from audio and text sources in multiple languages, including English, Arabic, and Mandarin Chinese, and provide English text versions via transcription and translation activities. GALE Distillation engines respond to formatted queries by searching information sources for relevant data and providing targeted English summaries of the relevant information.

Under the auspices of the GALE program, our team conducted a series of evaluations to characterize the information content, redundancy, and citation completeness of responses from GALE Distillation engines in comparison to responses from bilingual humans tasked with responding to focused, template-based queries. This section describes the evaluation framework we developed to collect statistics on distillation performance of machines and humans, the metrics we collected, and the salient results of the evaluations we conducted.

4.6.1.2. Evaluation Framework

The goal of the GALE evaluations was to compare the distillation performance of bilingual humans using non-GALE state-of-the-art tools to the performance of GALE Distillation engines. To accomplish this goal, BAE Systems' Advanced Information Technologies organization developed an evaluation framework that included realistic distillation tasks for both machines and humans, together with a methodology for comparing their performance. A crucial part of this framework was an approach to evaluation that did not require the existence of a "gold standard" in advance of the evaluations.

4.6.1.3. Task Definition

In each evaluation, GALE distillation engines were required to submit responses to a set of queries based on information contained in a preselected corpus of English, Mandarin, and Arabic in four genres (structured and unstructured, speech and text). The

queries conformed to 17 templates containing argument variables that ranged over events, topics, people, organizations, locations, and dates.

The GALE distillation engines and human distillers produced English-only snippets in response to these queries, which could consist of exact text extractions, translations, summarizations, or paraphrases of the source material. These output responses had to be clean and precise: irrelevant and redundant information incurred a penalty. Distillation engines and human distillers were also required to provide citations associating extracted snippets of information with the documents supporting them.

4.6.1.4. Machine Distiller Testing

In addition to extracted snippets of information, machine distillation engines (but not humans performing distillation) reported the following:

- *Snippet Chunks*, which were excerpts of snippets (also in English text) supported by source citations. Chunks did not need to be self-sufficient (interpretable independent of any other resource) and were allowed to contain non-contiguous text strings. In accordance with the XML specifications for the distillation responses, the Chunks included character offsets into the snippets in order to avoid ambiguity about which part of the snippet was being referenced.
- *Citations*, which were specific snippet chunks. Each citation indicated the chunk it supported and included the literal excerpt from the source document from which it originated (text string in the source language or audio time stamp), and an English transcription or translation for each foreign source.

Example

Query: DESCRIBE ATTACKS IN [Kosovo] GIVING LOCATION (AS SPECIFIC AS POSSIBLE), DATE, AND NUMBER OF DEAD AND INJURED.

Snippet 1: The UN police said the officer was killed late Sunday on the motorway between Leposavic and Mitrovica, some 55 kilometers (33 miles) north of the capital Pristina

Chunk 1.1: The UN police said the officer was killed late Sunday on the motorway

Citation 1.1.1: Menon became the first UN police man to die in the line of duty in Kosovo when he was ambushed late Sunday on a motorway in northern Kosovo.

Chunk 2.1: The officer was killed late Sunday some 55 kilometers (33 miles) north of the capital Pristina

Citation 2.1.1: Satish Menon, 43, from India's southern state of Kerala, was killed by sniper fire shortly before midnight Sunday while traveling in a U.N. police car near the village of Slatina, some 55 kilometers (33 miles) north of the capital, Pristina, police said

In order to incorporate new redundancy requirements distillation engines could mark potentially redundant context information by enclosing it inside of {curly braces}.

Snippet 2. {A United Nations policeman has been shot dead} in a sniper attack, {the UN said}, confirming the first fatality of an international officer while on duty here.

Such context information did not count against the Distillation engine if it was redundant, but also did not add to the system's recall score if it was new information.

4.6.1.5. Human Distiller Testing

A total of 18 participants completed queries for the GALE evaluation. Ten participants were bilingual (five Arabic/English speakers, five Mandarin/English speakers), and eight were English speakers with no Arabic or Mandarin language proficiency.

Humans used Copernic Desktop Search Version 2.02 for searching English text and Onfolio Desktop Search Version 2.02 for searching Arabic and Mandarin text. Both search tools required typing keywords into a textbox search field and returned a list of documents from the indexed corpora that contained those keywords. Windows Media Player was used for listening to audio files in all languages. Arabic and Mandarin speakers were also provided with translation dictionaries to support word lookup and translation between.

The humans' task was to construct answers to a series of questions by searching large volumes of text and audio data archived and indexed on the computer and to find relevant information (i.e., generate responses) by using the search tools provided. Participants' responses (i.e., snippets) could consist of direct quotations, translations (participants were required to translate all non-English information into English), or paraphrases of relevant information retrieved during their searches. They were instructed to resolve all ambiguous references and dates in their responses (paper calendars were provided for date resolution). Participants were also instructed to include source information for all responses. Participants were given a minimum of 30 minutes, and up to 60 minutes (at their discretion) to generate responses to each query.

Due to concerns that human distillers would not be able to locate relevant information for audio queries within a 60-minute time window, the following procedure was instituted to ensure non-blank responses to audio queries. For all audio queries, participants were initially provided with the same information (i.e., source date, activity date, etc.) provided to machine distillers, and were given up to 60 minutes to generate responses. After responding to all assigned queries, participants were provided with "hints" for each audio query that they had been assigned. Each hint consisted of the name of a single audio file and a 20-minute time window within that file that contained information relevant to the query.

Participants were instructed to generate new responses to the audio queries, using only information provided through the hints, regardless of whether they had earlier generated a response without any hint. Participants were given 30 additional minutes to respond to each query after receiving hints, and were instructed not to reference their no-hint responses in generating their hint responses. Thus, for all audio queries, each participant completed first a no-hint response and later a hint response. In a small number of cases, due to attrition prior to completion of hint responses, different participants completed the no-hint and hint responses for a given audio query.

4.6.1.6. Evaluation Methodology

In contrast to most previous evaluations of query answering systems, the GALE evaluations required evaluation of responses to very open-ended queries. There was no pre-existing set of "gold standard" answers against which the response of a given distiller

could be compared and scored. We therefore compared each distiller's response to a query to the total set of responses received for the query. Specifically, in determining what pieces of information a given distiller had *missed*, we looked only at information provided by other distillers. The scores of a distiller for a particular query were therefore *relative* scores, reflecting the degree to which the distiller provided relevant and specific responses to the queries as compared to other distillers. The details of computing these relative scores are spelled out in subsequent sections.

4.6.1.7. Annotation of Human and Machine Responses

After receiving all distiller responses, BAE evaluated each query individually.²² The first 500 words of each response were manually parsed into atomic facts or information units that answered the query in question, called *nuggets*. The nuggets, in turn, were mapped into equivalence classes of one or more nuggets, referred to as *nugs*. In Phase 2, BAE also performed citation checking to validate that the content of each nugget accurately reflected information in the source document.

For each query and distiller, BAE monitored the density of the distiller's response to determine how much of the response to parse for relevant information. If the density of the amount of relevant, non-redundant information of the first 500 words was less than 0.5, then no more of the distiller's response text was parsed. On the other hand, if the density equaled or exceeded 0.5, then an additional 200 words was parsed, nuggetized, and assigned to nugs. The density was then computed for the additional 200 words that had been parsed, and the 0.5 criterion applied. This process continued with additional blocks of 200 words until either the density was less than 0.5 or no more response text remained.

4.6.1.7.1. Nuggetization and Mapping to Nugs

Nuggetization consisted of decomposing each snippet returned by a distiller into nuggets (In some cases the entire snippet could be a nugget; more typically, however, a snippet consisted of multiple nuggets.)

Determining how to nuggetize a snippet was difficult, and different annotators seemed to have different intuitive notions of the appropriate granularity for nuggets. For example, consider the following snippet:

Snippet: Before receiving the war spending bill from Congress, the president will fly to the headquarters of the U.S. Central Command in Tampa, Florida

There are many ways that this sentence could have been broken down into smaller pieces of information. In order to ensure consistent annotations, we did not try to achieve the finest level of granularity, but rather developed the set of procedural rules listed below. Whereas, some nuggetization rules relied on linguistic criteria, others were based on end-user specified significance.

²² Annotators evaluating responses were not participating in generating human responses.

The first nuggetization rule was “A nugget is created out of each core verb and its arguments, where the maximal extent of the argument is always selected.” Since there are two core verbs in the italicized sentence above, two nuggets should be generated. The extent of the nugget in each example is indicated by double brackets:

Nugget 1. Before receiving the war spending bill from Congress, [[the president will fly to the headquarters of the U.S. Central Command]] in Tampa, Florida

Nugget 2. Before [[receiving the war spending bill from Congress, the president]] will fly to the headquarters of the U.S. Central Command in Tampa, Florida

The second rule said that “all temporal, locative, causative, and other types of modifiers of the verb constitute a separate nugget, including subordinate clauses, where the maximal extent of the modifier is always selected.” In this example, a nugget has been created out of the temporal clause:

Nugget 3: [[Before receiving the war spending bill from Congress]], the president will fly to the headquarters of the U.S. Central Command in Tampa, Florida

It is certainly possible to break this snippet down into much smaller pieces, for example, by breaking down noun phrases into ‘*the war spending bill*’ and ‘*bill from Congress.*’ This level of granularity, however, is often too fine for the task in question, where relevant nuggets serve as building blocks for answers to distillation queries. In order to provide a level of granularity corresponding to possible answers to the queries, as well as, to simplify the annotation task, the following rule was introduced: “Noun phrases are not decomposed into separate nuggets, unless they contain temporal, locative, numerical information, or titles.” These exceptions relate to the “significance” of created nuggets, where times and locations of events, as well as, quantitative information, were considered to be important pieces of information. Based on this rule, only one other nugget is created for the snippet above, which specifies the location of the headquarters of the U.S. Central Command:

Nugget 4: Before receiving the war spending bill from Congress, the president will fly to the headquarters of the U.S. Central Command [[in Tampa, Florida]]

Special rules were introduced for specific linguistic constructions, such as conjunctions or sentences with direct and indirect speech (see Babko-Malaya 2008 for discussion of nuggetization and relevancy in GALE Distillation).

Nuggets were further grouped together into larger units called “nugs.” A nug contains nuggets that convey similar information but that can differ with regard to the specificity of that information. For example, the nuggets *John was [[in Rome, Italy]]* and *John was [[in Italy]]* would belong to the same nug, however, the first nugget has a higher *degree of membership* in the nug than does the second, since it provides more specific information about John’s location. Degrees of membership in a nug are indicated by values between 0 and 1, inclusive. The information content of the nug itself is identified with the information content of the most specific nugget in the nug.

In the process of clustering nuggets into semantically equivalent classes, we distinguished between two types of nuggets: propositional nuggets, which introduced facts or propositions, and non-propositional nuggets, which introduce names, times, locations, or other entities.

In the case of propositional nuggets, we adopted the definition of ‘entailment’ from PASCAL Textual Entailment Challenge, which said “A entails B if the meaning of B can be inferred from the meaning of A, as it would typically be interpreted by people.” Using this notion of entailment, we defined degrees of nug membership for propositional nuggets as follows: “Nugget A has a degree of membership $D = 1$ in a nug if there is no other nugget in the nug which entails A. Nugget A has a degree of membership less than 1 in a nug if there is some other nugget B in the nug (not necessarily from the same distiller) that entails A and which is not entailed by A.” To ensure consistency, we assumed the following rule: “if nugget A entails nugget B and A is not entailed from B, then D is reduced by 0.2, independent on how different the nuggets are”:

[[The United Nations imposed sweeping sanctions on Iraq]], $D = 1$

[[The United Nations imposed sanctions on Iraq]], $D = 0.8$

[[Sanctions were imposed on Iraq]], $D = 0.6$

[[William Taylor is a defense lawyer]], $D = 1$

[[William Taylor is a lawyer]], $D = 0.8$

For non-propositional nuggets, i.e., nuggets which refer to an entity, a time interval, or a location, a different rule was introduced: “If a nug has 2 nuggets of differing membership, the degree of membership of the less specific one is reduced by 0.5. If a nug has 3 nuggets with different levels of specificity, then each time the degree of membership is reduced by 0.3. In the case of 4 nuggets – by 0.2, if there are more than 4 nuggets in a nug, D is reduced by 0.1”:

John Smith visited Italy [[on August 31, 1999]], $D = 1$

John Smith visited Italy [[in August 1999]], $D = 0.7$

John Smith visited Italy [[in 1999]], $D = 0.4$

John Smith visited [[Italy]], $D = 0.5$

John Smith visited [[Rome, Italy]], $D = 1$

Andrew Fastow faces [[31]] more charges, $D = 1$

Andrew Fastow faces [[about 30]] more charges, $D = 0.7$

[[Dozens]] more charges were filed against Andrew Fastow, $D = 0.4$

Nugs themselves were assigned a *degree of relevance*, reflecting how relevant they were to a given query. Degrees of relevance were likewise measured on a scale from 0 to 1, 1 being completely relevant. The reason for assigning degrees of relevance to nugs was that, although great pains were taken to spell out relevancy guidelines for each query template, there were inevitably borderline cases in which it was a matter of the annotator’s judgment how relevant a piece of information was to the given query.

Allowing the annotator to assign a degree of relevancy somewhere between 0 and 1 in such cases provided more flexibility than would have requiring the annotators to make categorical “relevant” or “not relevant” judgments that may have seemed arbitrary for each borderline case.

In order to ensure reliable and consistent annotation, each query was at least double annotated and adjudicated. The following metrics show inter-annotator agreement scores.

Relevancy agreement: reflected yes/no judgments on a snippet level. Every annotated snippet that had at least one nugget was considered relevant. Snippets from which no nuggets were created were considered irrelevant. Relevancy agreement counted the percentage of snippets that both annotators judged as relevant or irrelevant.

	Relevance Agreement	%Nugget Overlap
Audio	0.90	0.83
Text	0.89	0.82
Arabic	0.91	0.86
Chinese	0.85	0.72
English	0.91	0.85
Structured	0.89	0.79
Unstructured	0.89	0.80
Total	0.89	0.80

Table 4.30: Inter-Annotator Agreement Scores

Nugget Overlap: every relevant snippet was associated with a set of nuggets, which were represented as intervals over the text of the snippet. The overlap and difference between each pair of annotators’ nugget intervals were calculated, counting only ‘meaningful characters’, i.e., letters and numbers. Overlap was the number of meaningful characters that both annotators included in the extent of the nuggets, and difference (Diff) was the number of meaningful characters, which only one annotator included in the extent of the nuggets. The following formula was then used to determine percentage of nugget overlap: $\% \text{NuggetOverlap} = \text{Overlap} / (0.5 * \text{Diff} + \text{Overlap})$. Table 4.30 shows Phase 2 inter-annotator agreement scores across languages and different source types.

4.6.1.7.2. World Knowledge

Several new tags on nuggets were introduced in Phase 2 to deal with the fact that determinations of relevancy or degree of membership in a nug sometimes hinged upon world knowledge. For example, if the query was “Provide information on the Egyptian Islamic Jihad” and a distiller provided as part of the response the snippet “Dr. Ayman Zawahiri’s decade-long quest to weaponize and use Anthrax against US targets,” then determining the relevance of this snippet required the world knowledge that Zawahiri was associated with Egyptian Islamic Jihad. Since this information was not necessarily present in the corpus, it seemed unfair to penalize other distillers for missing this piece of information. On the other hand, the distiller returning this snippet had indeed returned a

relevant piece of information, so it seemed equally unfair to penalize that distiller by counting the snippet as irrelevant. In such cases, we tagged the extracted nugget with the “World Knowledge – Relevant” tag, indicating that determination of its relevancy required world knowledge not contained in the corpus. When a nugget was so tagged, the distiller providing the nugget was given credit for reporting it, but other distillers who failed to report equivalent information were not penalized.

Similarly, if world knowledge was required to determine whether one nugget was redundant with another (e.g., when that fact that two names referred to the same entity required empirical knowledge), then the “World Knowledge – Redundancy” tag was used for one of the nuggets. Nuggets with that tag did not trigger a penalty for redundancy.

4.6.1.7.3. Citation Checking

In year 2, we validated that nuggets were supported by the citations provided by the distiller. For each nugget, annotators first determined which part of the nugget was represented in the chunk for that nugget. (See Section 4.6.1.4 for a definition of “chunk”.) Annotators assigned a “chunk degree of membership” (cDM) between 0 and 1, which consisted of the content overlap between the nugget and the chunk. For example, if the nugget was fully contained in the chunk, cDM was 1. If none of the content of the nugget was contained in the chunk, cDM was set at 0. If no chunk either wholly or partially contained the nugget, then cDM was likewise 0.

For chunks which did support the nuggets (cDM>0), English-speaking and bilingual annotators checked citations (and documents, if needed) in order to validate that the information in the nugget that was contained in the chunk did accurately reflect information in the source document. Citation checkers assigned “degrees of support” (DS) ranging between 0 and 1 to estimate to what extent a given citation supported the information in the nugget that was represented in the chunk. In the following example, the nugget says that a certain event took place *on Wednesday, August 08, 2003*. The overlap between the nugget and the chunk, *August 2003*, is less specific than the information in the nugget, so the cDM is set as 0.5.²³ The degree of support (DS), on the other hand, is 1, given that *August 2003* is fully supported by the citation:

Nugget: At least one gunman opened machine-gun fire on teenagers swimming in a river in western Kosovo [[on Wednesday]], killing two Serb youths and injuring four others
Chunk: In August 2003 attackers gunned down two Serb youths
Citation: In August 2003, attackers gunned down two Serb youths while they were swimming in central Kosovo
 cDM = 0.5 and DS = 1

4.6.1.8. Metrics

The core metrics used in GALE evaluations measured recall, precision, and F-value for information retrieval and document citations. Another important metric that we computed was proficiency, which measured the proportion of the total information

²³ The guidelines for assigning chunk degree of membership followed the same rules as the guidelines for assigning nugget degree of membership.

provided by a given distiller. (The proficiency metric is not discussed in this section; see White, Hunter, and Goldstein 2008, for a definition of this metric.)

4.6.1.8.1. Information Retrieval Metrics

Information retrieval metrics were used to measure the performance of each distiller in returning information relevant to each query. To compute these metrics, we needed three numbers:

- The number of *right* nuggets
- The number of *wrong* nuggets
- The number of *missing* nuggets

The challenge was to compute these numbers in a context in which relevancy was taken to be a matter of degree, as was a nugget's membership in a nug.

Consider Table 4.31:

	1 - D	D
1 - R	_	#Wrong
R	#Missing	#Right

Table 4.31

For an arbitrary nug with relevance $R=[0,1]$ let $D=[0,1]$ be the *average* degree of membership of non-redundant nuggets from a given distiller. This nug contributes $(R)D$ to the distiller's #Right score, $(1 - R)D$ to the #Wrong score, and $R(1 - D)$ to the #Missing score. The final value of #Right, #Wrong, etc. includes the sum of each nug's contribution to that value.

Two additional values were needed to complete the computation of the contingency table entries. First, we needed to take into account that nuggets in text determined to be irrelevant should count as wrong; however, since such text was not nuggetized, the number of nuggets in it had to be estimated. We used as an estimate of the number of such nuggets $\max(0, C/40 - \text{\#Right})$, where C was the total number of non-blank characters in the response text. Second, we wanted to penalize redundant nuggets as wrong, so the #Wrong score had to include a term giving the number of redundant nuggets in any nug from a given distiller.

Given the entries in the generalized contingency table, we can compute the following information metrics (recall and precision):

$$\text{Recall} = \frac{\text{\#Right}}{\text{\#Missing} + \text{\#right}}, \quad \text{Precision} = \frac{\text{\#Right}}{\text{\#Wrong} + \text{\#right}}.$$

These are the two key metrics used to evaluate information retrieval performance and may be combined in the standard method to produce the single measure of information F-value

4.6.1.8.2. Citation Metrics

In evaluating the document support in a distiller's response text for a given query, the computations were analogous to those used in evaluating information retrieval performance.

As described in Section 4.6.1.4, source citations were associated with *chunks*, which were excerpts of snippets. A source citation for a chunk was considered to support a nugget if the chunk contained the nugget and the document confirmed the information content of the nugget. A nugget was allowed to be only partially contained in a chunk (some information in the nugget might have been missing from the chunk or the nugget might have contained information not in the chunk), so we quantified a nugget's *degree of membership*, DM, in the chunk by a value between 0 and 1. Each nugget *also* had a *degree of support* provided by the document associated with the chunk. We take degree of support, DS, to be a value between 0 and 1.

Thus for citations, the procedure for computing #Right was the following. We iterated through each chunk provided by the distiller, computed DS and DM for each nugget contained in the chunk, and took the sum of these values. A similar procedure was followed for #Wrong, except that the terms in the sum were in the form $(1 - DS)DM$.

The computation of the missing score required a somewhat different procedure from that described above. Since annotators did not examine the corpus for missing documents, in computing the missing increment for a nugget extracted from a given distiller's response, the annotator considered only documents provided by *other* distillers that were associated (via the chunks) to nuggets contained in the same nug as the nugget under evaluation. Documents cited by other distillers for nuggets in the nug but not cited by the given distiller counted toward the missing score for that distiller.

As in the case of information retrieval metrics, precision, recall, and F-value for document retrieval were computed using the same definitions, in terms of #Right, #Wrong, and #Missing.

4.6.1.8.3. GALE Evaluation Metric

The official GALE evaluation metric for Phase 2 combined the information and document retrieval metrics by weighting information *recall* by document *F-value* and then computing a weighted F-value using information precision and weighted information recall in place of simple information recall. The metric so computed was called *scaled F-value*. The rationale for this metric was that machine distillers should excel at recall because of their ability to process a large number of documents very rapidly; however, the value of retrieved information depended upon the documentation backing it up, so information recall should be weighted by the overall value (the F-value) of the retrieved documents. Perfect document retrieval should not diminish the weight placed on information recall, but missed or erroneous document citations should have the effect of lowering the value placed on information recall for a particular query.

4.6.1.8.4. Other Metrics

A wide range of additional metrics were computed during the performance evaluation. For a detailed discussion of the range of metrics, see White et. al. (2008).

4.6.1.9. Key Results in Human/Machine Comparison

In this section, we discuss results on the performance of machine distillers relative to humans. The results showed the ratio of the average machine performance to average human performance for a variety of key metrics, broken down by source type and language. We start with metrics by source type:

Information Metrics				
	Str. Text	Uns. Text	Str. Audio	Uns. Audio
Recall	0.77	1.21	0.82	0.40
Precision	0.66	0.66	0.67	0.58
F-value	0.50	0.78	0.49	0.28

Table 4.32: Relative information ratio scores by source type for structured and unstructured text and speech.

Document Metrics				
	Str. Text	Uns. Text	Str. Audio	Uns. Audio
Recall	0.88	1.40	0.78	0.33
Precision	0.97	1.07	1.11	1.03
F-value	0.93	1.30	0.83	0.37

Table 4.33: Relative document ratio scores by source type for structured and unstructured text and speech.

Combined Metrics				
	Str. Text	Unstr. Text	Str. Audio	Unstr. Audio
Scaled F-value ratio	0.54	0.80	0.52	0.25

Table 4.34: Combined relative metrics by source type.

The following observations can be drawn from the above tables.

- Machines performed better at document recall on Text than on Unstructured Audio
- Machine distillers significantly outperformed humans on both information and document recall for unstructured text
- Unstructured Audio files were most difficult for machine distillers

Next we consider scores broken down by language:

Information Metrics			
	English	Chinese	Arabic
Recall	0.76	0.53	1.08
Precision	0.70	0.77	0.48
F-value	0.60	0.34	0.53

Table 4.35: Relative information scores by language.

Document Metrics			
	English	Chinese	Arabic
Recall	0.69	0.61	1.30
Precision	1.02	1.07	1.04
F-value	0.74	0.63	1.27

Table 4.36: Relative document scores by language.

Combined Metrics			
	English	Chinese	Arabic
Scaled F-value ratio	0.59	0.36	0.61

Table 4.37: Combined relative metrics by language.

The scores by language tell us that Chinese was the most difficult language for machine distillers to process, while machine performance on Arabic exceeded that of humans. English was intermediate in difficulty, but for one metric – document precision – machine performance was better than that of humans.

Scores were also broken down by template type and argument type for templates. No correlation between template type and relative machine performance was detected. Many of the queries found to be difficult used names as their argument type, with names in Chinese audio files being particularly difficult.

4.6.2. Error Analysis and Future Directions for Distillation

Authors: Elizabeth Boschee, Marjorie Freedman, Roger Bock, John Graettinger, and Ralph Weischedel

4.6.2.1. Introduction

Evaluating Distillation performance is complicated because, 1) end-to-end system evaluation involves human assessment of system responses that include speech recognition, machine translation, response identification, and redundancy detection, 2) both response identification and redundancy detection are dependent on various natural language understanding components, including syntactic parsing, name recognition, co-reference (both within and across documents), relation and event detection, and 3) automatic scoring can only roughly approximate the human evaluators' judgments, due to the complex nature of the evaluation and the challenges of producing a complete answer key.

Detailed error analysis of the available evaluation results can therefore be highly valuable. The following analysis is based on AGILE's results in the Phase 2 Distillation evaluation and specifically examines BBN's contribution to the AGILE submission.

On many queries, AGILE performance was strong, sometimes producing responses twice or three times as comprehensive, accurate, and efficient as an average human's.

For this section, however, we have selected twenty of AGILE's lowest scoring query responses to analyze, under the assumption that looking specifically at system error can help identify potentially fertile ground for future research. The queries we have selected include examples from all four genres and all three languages.

We have identified five primary categories of errors: 1) the quality of recognized speech or translated text; 2) mistakes in co-reference which cause entities to be incorrectly linked or links between mentions to be missed; 3) failures of answer selection due to poor matching of complex topics; 4) failures of answer selection due to low feature coverage; and 5) highly redundant answers. While aspects of this analysis may be system-specific, the analysis does provide insights into which components of a Distillation System are most likely to cause errors. For each error category, we provide

some thoughts on how errors from this category could be addressed in the framework of the AGILE Distillation System.

4.6.2.2. Translation or Transcription

More than two-thirds of the evaluation queries called for responses solely from machine translation and/or transcription. For many of these queries, the translation/transcription (though imperfect) was of high enough quality to enable reasonably accurate Distillation. For instance, in response to a query about places visited by Hu Jintao, the system identified the following two responses from Chinese MT, despite the lower machine translation quality in the second example:

1. *Chinese President Hu Jintao's visit to the United States in September*
2. *Hu Jintao 2005 in September in New York to participate in the UN General Assembly during the summit*

However, sometimes the quality of the machine translation and transcription proved a significant stumbling block for Distillation; indeed, this was the case for eight of the lowest scoring queries analyzed here.²⁴ For seven of these queries, the specific problem was that a name in the query was missed in the translation or transcription of the correct answers. (For the eighth query, the MT responses were simply too garbled to be understood.)

This problem is an example of how evaluation in an end-to-end context can highlight problems of upstream processing which might not be immediately noticed except for their impact on a downstream component. The GALE MT evaluation does not distinguish between incorrect translation of names and the incorrect translation of non-name words. In Distillation, however, names are disproportionately important: 80% of the templates require the system to identify at least one named entity in the text, and even the ‘topic-based’ queries usually involve at least one name as a key component (e.g., “*List facts about [the death of Abu Musab al-Zarqawi]*”). The following table provides a few examples of instances where errors in name translation/transcription caused the Distillation System to miss responses. The underlined portion of the English text contains the words that should have been translated or transcribed as the missing name.

Missed Name	English Text
Abu Hamza al-Muhajir	On the Internet, said in a statement that the Iraqi anti-US armed forces leader <u>Zarqawi's successor Issa</u> al-personally...
Cher	“So could some simple padding in the helmet of marine save lives <u>share</u> thinks so and is now taking on the Pentagon
Hines Ward	the United States conduct rugby hero <u>Hans</u> about nine years

Table 4.38: Distillation Errors due to Missed Names

²⁴ For reference, these were queries GNG16, GNG17, GNG32, GNG43, GNG45, GNG46, GNG49, GNG53. More information about the Distillation evaluation and evaluation queries can be found in Distillation 5.1.

This problem was not unique to the AGILE team: of the eight queries analyzed here, only two had any correct responses returned by either the Rosetta or NIGHTINGALE System. Improving machine translation specifically for Distillation is, therefore, a program-wide necessity.

There are several potential avenues for improving Distillation performance over the current state-of-the-art in machine translation and transcription. Some improvements will come naturally, as machine translation and speech recognition continue to improve. However, focused work on improving the transcription and translation of names would be especially helpful. For instance, Ji and Grishman describe a method for explicitly identifying names in the source text and proposing translations to the MT system (Section 4.5.3). Parton *et al.* (2008) use information derived from user's queries to do focused identification and re-translation of names and deleted verbs (Section 4); Kingsbury and Chaudhari take a related approach in their use of query-time information to re-transcribe possible answer passages (Section 4.5.2). We expect there would be further gains by more closely integrating Distillation with the n-best or lattice output of machine translation and speech recognition.

An alternative path to be explored in parallel would be (1) translating the query (2) identifying the answer in the source language (3) translating the answer. (This path would address names that are mistranslated, but not those that are mistranscribed.) This path presents several challenges.

Step 1 requires very accurate English-to-source language translation models. For queries which are purely name-focused (e.g., provide information about [PERSON]), a model that emphasized transliteration might suffice, however for topic-based queries (where names can still be critical), full translations of the topics are necessary.

Step 2 requires high quality language understanding technology in the source language. While the Automatic Content Extraction evaluation has included Arabic and Chinese evaluations for the past several years, syntactic parsing and extraction particularly in Arabic have lagged behind English performance.

Step 3 would be trivial if the Distillation System were simply to return the answer as originally translated by the machine translation system. However, this strategy will not always be successful. The goal of the three-step process is to improve performance in cases where machine translation quality has prevented the English Distillation System from finding a relevant response. If a machine translation is poor enough to confuse the language understanding components of the Distillation System, it is also likely to confuse an evaluator reading a response drawn from that same translation. For instance, consider a query like "*List facts about [tips in Chinese restaurants]*". In response to this query, the system might find a relevant passage in the source language that should be translated as "*In China, notices are posted in many restaurants prohibiting attendants from accepting tips from customers*". However, if the system returns the machine translation "*many Chinese restaurants were pasted notices to that the attendant received the tip of the guests*", this may be considered incorrect by an evaluator who rightly thinks that it omits the real meaning of the passage. Instead, it will be necessary to build on the related work described above to further take advantage of the Distillation System's understanding of both the query and source language response to improve answer translation quality.

4.6.2.3. Co-Reference

To successfully answer a question about a person, organization or country, the Distillation System should identify all mentions of that entity, even when the mention uses an alias, alternative spelling, or pronoun.

Name in Query	Mistake
the Egyptian Islamic Jihad	Missed link: “al-Jihad al-Islami”
the Egyptian Islamic Jihad	Spurious link: “Hezbollah”
Awad Hamed al-Bandar	Missed link: “Awwad Ahmad al-Bandar”
New York Police Department	Missed link: “New York cops”
European Union	Spurious link: “The UN Secretary-General Kofi Annan advised on holding an administrative reform complete to <u>the organization [European Union]</u> ” (“ <i>the organization</i> ” <i>does not refer to the EU</i>)

Table 4.39: Distillation Errors due to Co-Reference Mistakes that led to errors in the evaluation.

For most queries the AGILE system was able to do this successfully. However, for three of the lowest-scoring queries, we identified the primary source of error as a mistake in co-reference.

BBN relied on SERIF’s co-reference for links between names, pronouns, and nominal mentions within a single document. Continued improvements to within document co-reference—including incorporating the expanded notion of co-reference available through OntoNotes — will improve Distillation. However, errors in within-document co-reference frequently require expanded world knowledge to be solved. For example, the incorrect link between ‘organization’ and ‘European Union’ could be made less likely if the co-reference model were aware that Kofi Annan is more likely to be associated with the UN than the EU.²⁵ Initial results of using cross-document co-reference information to improve within-document co-reference decisions for named entities were promising in the ACE 2008 evaluation (Baron and Freedman, 2008).

For Distillation, we addressed one aspect of the problem of cross-document co-reference by using a set of features to identify name variation links, for example identifying that *Al Qaeda* and *Al Qaida* refer to the same organization. These links allowed the system to answer a query about named entity X using passages mentioning only X’ but did not disambiguate document entities that might share those names X or X’, i.e., distinguishing between all people named *John Smith*. Instead we relied on the context of the query to guide the system to the correct documents. For example, given the query *Describe the prosecution of [John Smith]*, the system would naturally focus only on *John Smiths* who were mentioned as trial defendants.

Since the Phase 2 evaluation, BBN has integrated the name variation component into a full named entity cross document co-reference component (Baron and Freedman 2008). Experiments with cross document co-reference showed that adding context-based features (e.g., premodifiers from the extraction system, topic predictions) to the name

²⁵ The slightly garbled machine translation also contributed to this error.

variation component improved cross-document co-reference. For example in the Phase 2 component, the names *Koufodinas* and *Koufontianas* were not linked (though the second is just a misspelling of the first). Both names are mentioned in conjunction with events related to terrorism in Greece. With the addition of context-based features, the system correctly linked these two names.

Using language-specific variants of edit distance could also improve performance. For example, the variation between ‘ui’ and ‘u’ in a Chinese name usually indicates two separate entities (e.g. Suizhou and Suzhou). However, the same variation in a Hindi name is usually just a misspelling (e.g., Kastruirangan and Kastrurangan). One challenge with this approach would be automatically identifying the linguistic origins of a name not published in its native language.

4.6.2.4. Proposition Tree Matching Errors

For four of the lowest-scoring queries in the evaluation set, we identified semantic matching as the primary source of error. In other words, the system did not correctly match a user-specified topic to the relevant passages (which were potentially expressed in very different words).²⁶ For templates with event- and topic-based slots, the BBN system uses ‘proposition trees’ to identify passages that are syntactically and semantically similar to the topic or event. Details about proposition trees and the associated matching algorithm are described in Section 4.4.4.3.

Briefly, a proposition tree is a representation of text content that is designed to capture the normalized predicate-argument structure of a sentence (*who did what to whom*). For Distillation, the BBN system generates proposition trees for both the query and all possibly relevant response documents. It then augments these trees with synonyms and co-reference information, and uses a matching algorithm over the resulting trees to identify document passages relevant to a particular query, even when the passages refer to the topic using different words or syntactic structure. For instance, the system will use the following proposition trees (shown pre-augmentation) to match the query topic “the arrest of Abu Abbas in Baghdad” to the document passage “he was captured in the capital”:

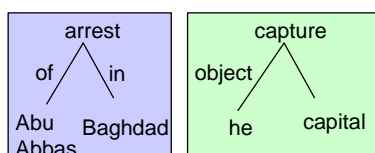


Figure 4.28: Proposition Trees

Although the proposition tree matching algorithm enables the system to detect relevant passages with greater recall and precision than surface-level strategies, the proposition tree matching algorithm can make two kinds of mistakes: it can either be too lenient (causing irrelevant responses) or too conservative (causing missed responses).

²⁶ Queries GNG06, GNG23, GNG29, GNG34

Irrelevant responses are usually the result of the algorithm not identifying the most important words in the topic. For example, in answering the query “What connections are there between [World Cup games] and [stock markets]?”, BBN provided the response “*The government to rescue the original pattern of the game, as well as, several times in the game in the formation of the rules of the game, ... the company’s basic law of the market...*” The matching algorithm did not know that it should not have expanded to games beyond the “World Cup”, i.e., that “World Cup” was a very important modifier. The current matching algorithm weights words primarily by their syntactic position: predicates and roots are more important than the words that modify them. At BBN, we plan to address this issue by introducing more complex features for weighting words, perhaps placing more weight on certain types of modifiers (e.g., words that are capitalized, or words that are less frequent).

Missed responses are frequently the result of the algorithm not understanding when context from one sentence in a document carries over to a neighboring sentence. For example, for the query “List facts about [the signing of the Adam Walsh Child Protection and Safety Act],” the system only reported the underlined portion of the following relevant transcription: “*John Walsh who courses a host of America’s most wanted his son Adam was abducted exactly 25 years ago today. And was found murdered weeks later and today the president signed the Adam Walsh Child Protection Safety Act*”. At BBN, we plan to improve semantic matching performance in such cases through more sophisticated modeling of topic flow and segmentation over documents.

4.6.2.5. Low Feature Coverage

For query templates that do not rely on open-ended topics and events, the BBN Distillation System frequently relies on specific lexically- and syntactically-driven patterns that produce features for the answer selection process. These patterns, based on linguistic elements produced by SERIF (e.g., entity mentions, relations, events, and propositions), are typically very successful at capturing relevant answers for these entity-based queries. However, when answers appear in atypical contexts (e.g., amid garbled machine translation), the Distillation System sometimes misses responses due to lack of pattern coverage. We determined that the poor performance on three of the lowest scoring queries was primarily the result of answers that were not covered by our features.²⁷ The two examples below illustrate instances where BBN missed responses (underlined) because of this problem:

²⁷ Queries GNG31, GNG41, GNG42

- (A) *Where has [Robert B. Zoellick] been and when?*
- ... Robert Zoellick told an audience this week in Khartoum, Sudan's capital.
 - The United States opened a consulate in Juba, in southern Sudan. Robert B. Zoellick, the deputy secretary of state, officiated.
 - As three white helicopters carrying Robert B. Zoellick and his traveling party began descending over the Kalma refugee camp...
- (B) *Produce a biography of [Jacques Rene Chirac]*
- French President Jacques Chirac at the 72-year-old
 - CHIRAC-72-year-old CHIRAC-72-year-old
 - French presidential office announced Friday, French President Jacques vascular problems led minor problems. from 2:hospital one week

In the missed responses to (A), the locations are not easily syntactically connected to Zoellick or his actions. In the missed responses to (B), the machine translation led to phrases that are (generally) understandable by a person but which BBN's Distillation System was not able to understand.

We believe there are several avenues for improving feature coverage. By partially automating feature proposal, we could increase the number of features used to identify answers without additional effort. We also plan to use the GALE semantic annotations to extend the coverage of lexically-based patterns. For example, automatic word-sense disambiguation, combined with an ontology linking word senses, can be used to expand the word 'support' to 'endorse' and 'back' in appropriate contexts. Work that identifies answers in the source language text will also be helpful in identifying answers such as "French President Jacques Chirac at the 72-year-old," where the machine translation is understandable, but not covered by a pattern.

4.6.2.6. Redundancy

The GALE Distillation task involves not only identifying relevant responses to a query, but also compressing redundant information into a non-redundant response. For instance, when asked about the activities of Abu Musab al-Zarqawi, the BBN system recognized that "*al-Zarqawi, who has very definitely been involved in the kidnapping and beheading of foreigners*" repeats information found in "*Al-Zarqawi has claimed responsibility for numerous attacks on U.S. and Iraqi forces, along with the kidnappings and the beheadings of several foreigners, including Americans.*"

Given the complexity and sub-sentence granularity of redundancy, this component was particularly challenging to evaluate outside of the full GALE evaluation. We identified two queries for which the primary cause of failure was redundant information that was not removed from the response.²⁸ The algorithm used to identify redundant information combines lexical and syntactic information to identify chunks of an answer that are redundant. For machine translated text, the syntactic parse is typically degraded. Consequently, it is more difficult to identify which chunks are redundant, or which chunks are the most important (typically those closest to the root of the parse). At BBN, we will address this problem by exploring ways to decouple redundancy detection from the parse structure, making the process more robust to syntactic degradation. We will also

²⁸ Queries GNG19, GNG35

add additional sources of lexical redundancy from the GALE semantic annotations. We expect this to help identify the redundancy of snippets like these (both returned in response to a query about Airbus): “*A.380 generation Airbus postponed delivery ceremony*” and “*Boeing’s rival Airbus company has announced the extension of the Airbus a.380 delivery period*”. In this case the system has failed to recognize the similarity between the core concepts “announced the extension” and “postponed”; better and broader lexical features will help here.

4.6.2.7. Conclusion

We have discussed five primary sources of error for the GALE Distillation task. The most significant contributor numerically is the translation and transliteration of names. Other primary contributors are co-reference error, answer selection error from poor matching of complex topics, answer selection error from low feature coverage, and redundancy.

To improve Distillation over imperfect machine translation and transcription, we have suggested pursuing Distillation work in the source language; the main challenge here is not answer identification (though this can be difficult as well), but discovering how to use the identification of source language answers to improve overall translation so that those answers can be understood in English.

To improve co-reference, we have discussed ongoing work at BBN using context-specific features and other improvements to a cross-document co-reference system. The integration of additional language understanding resources (e.g., OntoNotes) will also help improve co-reference performance. We expect these resources could also help minimize answer selection error, for instance by allowing the system to automatically expand feature coverage with high precision, or by providing additional information to the semantic matching algorithms.

Finally, redundancy removal is by itself a significant research task. Planned algorithms for redundancy detection will better balance the need to incorporate and account for syntactic structure with the reality of operating over imperfect text with degraded parses. Incorporating additional semantic features gleaned from the GALE semantic annotations (OntoNotes) will improve the accuracy of redundancy detection. Developing automatic evaluation techniques for this component would improve our ability to monitor progress.

In sum, we have used the results of the GALE Phase 2 Distillation evaluation to identify significant sources of error and propose possible research directions to address these errors. The end goal should be improved performance across the board for Distillation.

References

- Ajmera, J. and C. Wooters. 2003. A Robust Speaker Clustering Algorithm. *Proceedings of the IEEE workshop on Automatic Speech Recognition and Understanding*.
- Al-Onaizan, Y. and K. Knight. 2002. Translating Named Entities Using Monolingual and Bilingual Resources. *Proceedings of the Meeting of the Association for Computational Linguistics*.
- Artiles, J., S. Sekine, and J. Gonzalo. 2008. Web People Search - Results of the First Evaluation and the Plan for the Second. *Proceedings of the International World Wide Web Conference*.
- Babko-Malaya, O. 2008. Annotation of Nuggets and Relevance in GALE Distillation Evaluation. *Proceeding of the International Conference on Language Resources and Evaluation*.
- Banerjee, S. and A. Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. *Proceedings of Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization at the Meeting of the Association for Computational Linguistics*.
- Bar-Haim, R., I. Dagan, W. Dolan, L. Ferro, D. Giampiccolo, B. Magnini, and I. Szpektor. 2006. The Second PASCAL Recognising Textual Entailment Challenge. *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.
- Baron, A. and M. Freedman. 2008. Who is Who and What is What: Experiments in Cross Document Co-Reference. *Empirical Methods in Natural Language Processing*. 274–283.
- Bartell, B., G. Cottrell, and R. Belew. 1994. Automatic Combination of Multiple Ranked Retrieval Systems. *Proceedings of 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. 173–181.
- BeanShell. 2009. <http://www.beanshell.org/>
- Bikel, D. M. 2004. A Distributional Analysis of a Lexicalized Statistical Parsing Model. *Proceedings of the Conference on Empirical Methods in Natural Language Processing*.
- Bikel, D. M., S. Miller, R. Schwartz, and R. Weischedel. 1997. Nymble: a High Performance Learning Name-Finder. *Proceedings of ANLP* pp. 194–201.
- Boschee, E., R. Weischedel, and A. Zamanian. 2005. Automatic Information Extraction. *Proceedings of the International Conference on Intelligence Analysis*.
- Boschee, E., V. Punyakanok, and R. Weischedel. 2008. An Exploratory Study Towards Machines that Learn to Read. *Proceedings of AAAI'08 BICA*.
- Brill, E. 1992. A Simple Rule Based Part of Speech Tagger. *Proceedings of the Third Conference on Applied Natural Language Processing*.

- Brill, E., and R. Moore. 2000. An Improved Error Model for Noisy Channel Spelling Correction. *Proceedings of the Meeting of the Association for Computational Linguistics*. 286–293.
- Brown, P. E, V. J. Della Pietra, P. V. deSouza, J. C. Lai and R. L Mercer. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, 18(4):467–480.
- Cabezas, C., and P. Resnik. 2005. Using WSD Techniques for Lexical Selection in Statistical Machine. *Translation Technical report CS-TR-4736 / LAMP-TR-124 / UMIACS-TR-2005-42*.
- Carpuat, M., and D. Wu. 2007. Context-Dependent Phrasal Translation Lexicons for Statistical Machine Translation. *Proceedings of Machine Translation Summit XI*.
- Chambers, N., D. Cer, T. Grenager, D. Hall, C. Kiddon, W. MacCartney, M. de Marneffe, D. Ramage, E. Yeh, and C. Manning. 2007. Learning Alignments and Leveraging Natural Logic. *Proceedings of the Meeting of the Association for Computational Linguistics-PASCAL Workshop on Textual Entailment and Paraphrasing*.
- Chan, H. Y., and P. C. Woodland. 2004. Improving Broadcast News Transcription by Lightly Supervised Discriminative Training. *Proceedings of the ICASSP*.
- Charniak, E., and K. Knight. 2003. Syntax-Based Language Models for Machine Translation. *Proceedings of Machine Translation Summit IX*.
- Charniak, E. 2000. A Maximum-Entropy-Inspired Parser. *The NAACL*.
- Chaudhari, U., and M. Picheny. 2007. Improvements in Phone Based Audio Search via Constrained Match with High Order Confusion Estimates. *Proceedings of the IEEE Automatic Speech Recognition and Understanding Workshop*.
- Chen, S. F. 2003. Conditional and Joint Models For Grapheme-to-Phoneme Conversion. *Proceedings of the Eurospeech Conference*.
- Church, K. 1988. A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. *Proceedings of the Second Conference on Applied Natural Language Processing*. 136–143.
- Cohen, W., P. Ravikumar, and S. Fienberg. 2003. A Comparison of String Distance Metrics for Name-Matching Tasks. *Proceedings of the IIWeb Workshop*.
- Collins, M. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. *EMNLP*. 1-8.
- Collins, M. 2003. Head-Driven Statistical Models for Natural Language Parsing. *Computational Linguistics* 29: 589-637.
- Condoravdi, C., R. Crouch, V. de Pavia, R. Stolle, and D. Bobrow. 2003. Entailment, Intensionality, and Text Understanding. *Workshop on Text Meaning*.
- Crammer, K., J. Keshet, S. Shalev-Schwartz, and Y. Singer. 2006. Online Passive-Aggressive Algorithms. *Journal of Machine Learning Research* 7: 551–585.

- Cristianini, N., and J. Shawe-Taylor. 2000. An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods. *Cambridge University Press*.
- Dagan, I., O. Glickman, B. Magnini. 2006. The PASCAL Recognising Textual Entailment Challenge. *MLCW 2005, LNAI 3944*: 177–190.
- Davis, J., and M. Goadrich. 2006. The Relationship between Precision-Recall and Roc Curves. *Proceedings of the 23rd International Conference on Machine Learning*. 233–240.
- De Marneffe, M., A. N. Rafferty, and C. Manning. 2008. Finding Contradictions in Text. *Proceedings of the ACL*.
- De Marneffe, M., W. MacCartney, and C. Manning. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. *Proceedings of the 5th International Conference on Language Resources and Evaluation*.
- De Marneffe, M., T. Grenager, W. MacCartney, D. Cer, D. Ramage, C. Kiddon, and C. Manning. 2007. Aligning Semantic Graphs for Textual Inference and Machine Reading. *Proceedings of the AAAI Spring Symposium at Stanford*.
- Elming, J. 2006. Transformation-Based Correction of Rule-Based MT. *Proceedings of the European Association for Machine Translation Conference*. 219–226.
- Emms, M. 2006. Variants of Tree Similarity in a Question Answering Task. *Proceedings of the Linguistic Distances Workshop Held in Conjunction with the International Conference on Computational Linguistics*. 100–108.
- Farber, B., D. Freitag, N. Habash, and O. Rambow. 2008. Improving NER in Arabic Using a Morphological Tagger. *Proceedings of The International Conference on Language Resources and Evaluation*.
- Ferrández, S., A. Toral, O. Ferrández, A. Ferrández, and R. Muñoz. 2007. Applying Wikipedia's Multilingual Knowledge to Cross Lingual Question Answering. *Proceedings of the 12th International Conference on Applications of Natural Language to Information Systems*. 352–363.
- Fiscus, J. 1997. A Post-Processing System to Yield Reduce Word Error Rate: Recognizer Output Voting Error Reduction (ROVER). *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding*.
- Florian, R., H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N. Nicolov, and S. Roukos. 2004. A Statistical Model for Multilingual Entity Detection and Tracking. *Proceedings of the Meeting of the ACL: Human Language Technologies*. 1–8.
- Freitag, D., and S. Khadivi. 2007. A Sequence Alignment Model Based on the Averaged Perceptron. *Proceedings of the EMNLP and the Conference on Natural Language Learning*.
- Freitag, D. 2004. Trained Named Entity Recognition Using Distributional Clusters. *Proceedings of the EMNLP*.

- Freund, Y., and R. E. Schapire. 1999. Large Margin Classification Using the Perceptron Algorithm. *Machine Learning* 37(3): 277–296.
- Gales, M. J.F., D. Y. Kim, P. C. Woodland, H. Y. Chan, D. Mrva, R. Sinha, and S. E. Tranter. 2006. Progress in the CU-HTK Broadcast News Transcription System. *IEEE Transactions on Audio, Speech, and Language Processing* 14(5): 1513–1525.
- Giampiccolo, D., I. Dagan, B. Magnini, and W. Dolan. 2007. The Third PASCAL Recognizing Textual Entailment Challenge. *Proceedings of the ACL PASCAL Workshop on Textual Entailment and Paraphrasing*.
- Gillick, D., D. Hakkani-Tür, and M. Levit. 2008. Unsupervised Learning of Edit Parameters for Matching Name Variants. *Proceedings of ICSLP*.
- Gonzalo, J., and D. Oard. 2004. iCLEF 2004 Track Overview: Pilot Experiments in Interactive Cross-Language Question Answering. *Multilingual Information Access for Text, Speech and Images, Fifth Workshop of the Cross-Language Evaluation Forum Revised Selected Papers Series*, LNCS (3491): 310–322.
- Grishman, R., D. Westbrook, and A. Meyers. 2005. NYU’s English ACE 2005 System Description. *Technical Report 05-019, NYU Proteus Project*.
- Grishman, R., D. Westbrook, and A. Meyers. 2005. NYU’s English ACE 2005 System Description. *Proceedings of the Automatic Content Extraction Evaluation Workshop*.
- Hakkani-Tür, D., and G. Tur. 2007. Statistical Sentence Extraction for Information Distillation. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*.
- Hakkani-Tür, D., G. Tur, and A. Chotimongkol. 2005. Using Syntactic and Semantic Graphs for Call Classification. *Proceedings of the Workshop on Feature Engineering for Machine Learning in Natural Language Proceedings of the Meeting of the Association for Computational Linguistics*.
- Harabagiu, S., A. Hickl, and F. Lacatusu. 2006. Negation, Contrast, and Contradiction in Text *Proceedings of the Twenty-First National Conference on Artificial Intelligence*.
- Hastie, T., and R. Tibshirani. 1990. *Generalized Additive Models*. London: Chapman and Hall.
- Hillard, D., B. Hoffmeister, M. Ostendorf, R. Schülter, and H. Ney. 2007. iRover: Improving System Combination with Classification. *Proceedings of the Meeting of the Association for Computational Linguistics: Human Language Technologies*. 65–68.
- Hobbs, J. R. 1977. Resolving Pronoun References. *Readings in Natural Language Processing*, B. Grosz et al. eds., 1986. Morgan Kaufmann.
- Huang, F., S. Vogel, and A. Waibel. 2004. Improving Named Entity Translation Combining Phonetic and Semantic Similarities. *Proceedings of the Meeting of the Association for Computational Linguistics: Human Language Technologies*.

- Huang, L., K. Knight, and A. Joshi. 2006. Statistical Syntax-Directed Translation with Extended Domain of Locality. *Proceedings of the Meeting of the Association for Machine Translation of the Americas*.
- Ji, H., and R. Grishman. 2006. Analysis and Repair of Name Tagger Errors. Proceedings of the International Conference on Computational Linguistics / Association for Computational Linguistics Conference.
- Ji, H., and R. Grishman. 2007. Collaborative Entity Extraction and Translation. Proceedings of the International Conference on Recent Advances in Natural Language Processing.
- Ji, H., and R. Grishman. 2009. Collaborative Entity Extraction and Translation. Recent Advances in Natural Language Processing. Amsterdam and Philadelphia: John Benjamins Publishers.
- Ji, H., R. Grishman, and W. Wang. 2008. Phonetic Name Matching For Cross Lingual Spoken Sentence Retrieval. Proceedings of the IEEE-Association for Computational Linguistics Spoken Language Technology Conference.
- Joachims, T. 1999. Making Large-Scale SVM Learning Practical. In *Advances in Kernel Methods - Support Vector Learning*. B. Scholkopf, C. Burges, and A. Smola, eds. MIT Press.
- Kalmar, P., and M. Blume. 2007. Web Person Disambiguation via Weighted Similarity of Entity Contexts. Proceedings of the Association for Computational Linguistics Workshop on SemEval.
- Kernighan, M. D., K. W. Church, and W. A. Gale. 1990. A spelling correction program based on a noisy channel model. Proceedings of the International Conference on Computational Linguistics. 205–210.
- Kingsbury, P., M. Palmer, and M. Marcus. 2002. Adding Semantic Annotation to the Penn TreeBank. Proceedings of the Human Language Technology Conference.
- Klein, D., and C. D. Manning. 2003. Accurate Unlexicalized Parsing. Proceedings of the 41st Annual Meeting of the Association of Computational Linguistics.
- Klementiev, A., and D. Roth. 2006. Named Entity Transliteration and Discovery from Multilingual Comparable Corpora. Proceedings of the Meeting of the Association for Computational Linguistics: Human Language Technologies.
- Knight, K. and I. Chander. 1994. Automated Post-Editing of Documents. Proceedings of the National Conference on Artificial Intelligence (AAAI).
- Kouylekov, M., and B. Magnini. 2005. Tree Edit Distance for Textual Entailment. *Proceedings of the Recent Advances in Natural Language Processing Conference*.
- Kurohashi, S., T. Nakamura, Y. Matsumoto, and M. Nagao. 1994. Improvements of Japanese Morphological Analyzer JUMAN. Proceedings of The International Workshop on Sharable Natural Language Resources. 22–28.

- Lavie, A. 2008. A General Search-Based Syntax-Driven Framework for Machine Translation. *Proceedings of the Conference on Intelligent Text Processing and Computational Linguistics LNCS 4919*: 362-375.
- Lee, L. 1999. Measures of Distributional Similarity. *Proceedings of the Meeting of the Association for Computational Linguistics*. 25–32.
- Levit, M., D. Hakkani-Tür, G. Tur, and D. Gillick. 2007. Integrating Several Annotation Layers for Statistical Information Distillation. *Proceedings of the IEEE workshop on Automatic Speech Recognition and Understanding*.
- Levit, M., E. Boschee, and M. Freedman. 2007. Selecting On-Topic Sentences from Natural Language Corpora. *Proceedings of the InterSpeech Conference*. 2793-2796.
- Lin, C. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. *Proceedings of the Workshop on Text Summarization Branches Out, Post-Conference Workshop of the Meeting of the Association for Computational Linguistics*.
- Lin, C. L., and E. Hovy. 2003. Automatic Evaluation of Summaries Using N-gram Co-occurring Statistic. *Proceedings of the Human Technology Conference*.
- Lin, D., S. Zhao, B. Van Durme, and M. Paşca. 2008. Mining Parenthetical Translations from the Web by Word Alignment. *Proceedings of the Meeting of the Association for Computational Linguistics*.
- Luo, X., A. Ittycheriah, H. Jing, N. Kambhatla, and S. Roukos. 2004. A Mention-Synchronous Co-reference Resolution Algorithm Based on the Bell Tree. *Proceedings of the Meeting of the Association for Computational Linguistics Main Volume*:135– 142.
- Ma, J., and S. Matsoukas. 2005. Improvements to the BBN RT04 Mandarin Conversational Telephone Speech Recognition System. *Proceedings of the Inter-Speech Conference*.
- Ma, W. Y., and K. McKeown. 2009. Where's the Verb? Correcting Machine Translation During Question Answering. *Proceedings of the Meeting of the Association for Computational Linguistics*.
- MacCartney, W., T. Grenager, M. de Marneffe, D. Cer, and C. D. Manning. 2006. Learning to Recognize Features of Valid Textual Entailments. *Proceedings of the Meeting of the Association of Computational Linguistics*.
- Marcus, M. P., B. Santorini, and M. A. Marcinkiewicz. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics* 19(2):313–330.
- Mausser, A., D. Vilar, G. Leusch, Y. Zhang, and H. Ney. 2007. The RWTH Machine Translation System for IWSLT 2007. *In Proceedings of the International Workshop on Spoken Language Translation*.
- McCallum, A., K. Bellare, and F. Pereira. 2005. A Conditional Random-Field for Discriminatively-Trained Finite-State String Edit Distance. *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*.

- McCullagh, P. and J. A. Nelder. 1989. Generalized Linear Models. *Chapman and Hall, London*. NIST, ftp://jaguar.ncsl.nist.gov/current_docs/sctk/doc/sclite.Htm
- Miller, S., H. Fox, L. Ramshaw, and R. Weischedel. 2000. A Novel Use of Statistical Parsing to Extract Information from Text. *Proceedings of the Meeting of the Association for Computational Linguistics*. 226-233.
- Miller, S., J. Guinness, and A. Zamanian. 2004. Name Tagging with Word Clusters and Discriminative Training. *Proceedings of the Meeting of the ACL:HLT*. 337-342.
- Nairn, R., C. Condoravdi, and L. Karttunen. 2006. Computing Relative Polarity for Textual Inference. *Proceedings of the International workshop on Inference in Computational Semantics*.
- Nanda K. 2004. Combining Lexical, Syntactic, and Semantic Features with Maximum Entropy Models for Extracting Relations. *Proceedings of the Meeting of the Association for Computational Linguistics*. 22.
- NIST. 2000. http://www.itl.nist.gov/iad/mig/tests/ctr/2000/h5_2000_v1.3.html
- Och, F. J. 2003. Minimum Error Rate Training for Statistical Machine Translation. *Proceedings of the Meeting of the Association for Computational Linguistics*.
- Palmer, M., D. Gildea and P. Kingsbury. 2005. The Proposition Bank: A Corpus Annotated with Semantic Roles. *Computational Linguistics* 31(1).
- Papineni, K., Salim R., Todd W., and W. J. Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. *Proceedings of the Meeting of the ACL*. 311-318.
- Parton, K., K. McKeown, J. Allan, and E. Henestroza. 2008. Simultaneous Multilingual Search for Translingual Information Retrieval. *Proceedings of the Conference on Information and Knowledge Management*.
- Povey, D., B. Kingsbury, L. Mangu, G. Saon, and H. Soltau. 2005. fMPE: Discriminatively Trained Features for Speech Recognition. *Proceedings of the ICASSP*.
- Pradhan, S., W. Ward, K. Hacioglu, J. Martin, and D. Jurafsky. 2004. Shallow Semantic Parsing using Support Vector Machines. *Proceedings of the Meeting of the ACL:HLT*.
- Punyakanok, V., D. Roth, W. T. Yih, and D. Zimak. 2004. Natural Language Inference via Dependency Tree Mapping: An Application to Question Answering. *Computational Linguistics*.
- Ramshaw, L., E. Boschee, S. Miller, R. Stone, R. Weischedel, and A. Zamanian. 2001. Experiments in Multi-Modal Automatic Content Extraction. *Proceedings of the HLT Conference*. 1-5.
- Ravichandran, D. and E Hovy. 2002. Learning surface text patterns for a question answering system. *Proceedings of the ACL*. 41-47.
- Rosti, A., N. Ayan, B. Xiang, S. Matsoukas, R. Schwartz, and B. Dorr. 2007. Combining Outputs from Multiple Machine Translation Systems. *Proceedings of the Meeting of the ACL:HLT*. 228-235.

- Roth, R., O. Rambow, N. Habash, M. Diab, and Cynthia Rudin. 2008. Arabic Morphological Tagging, Diacritization, and Lemmatization Using Lexeme Models and Feature Ranking. *Proceedings of ACL*.
- Sanchez-Graillet, O., and M. Poesio. 2007. Discovering Contradiction Protein-Protein Interactions in Text. *Proceedings of the Conference on Biological, Translational, and Clinical Language Processing (BioNLP)*.
- Schapire, R. E., and Y. Singer. 1999. Improved Boosting Algorithms Using Confidence-Rated Predictions. *Machine Learning* 37(3): 297–336.
- Si, L., and J. Callan. 2005. Modeling Search Engine Effectiveness for Federated Search. *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*. 83-90.
- Simard, M., C. Goutte, and P. Isabelle. 2007. Statistical Phrase-Based Post-Editing. *Proceedings of the Meeting of the ACL:HLT*.
- Singla, A., and D. Hakkani-Tür. 2008. Cross Lingual Sentence Extraction for Information Distillation. *Proceedings of the ICSLP (Inter-speech)*.
- Soltau, H., F. Metze, C. Fugen, and A. Waibel. 2001. A One-Pass Decoder Based on Polymorphic Linguistic Context Assignment. *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*.
- Soltau, H., G. Saon, B. Kingsbury, J. Kuo, L. Mangu, D. Povey, M. Omar, and G. Zweig. 2007. The IBM 2006 GALE Arabic ASR System. *Proceedings of the ICASSP*.
- Sproat, R., T. Tao, and C. X. Zhai. 2006. Named Entity Transliteration with Comparable Corpora. *Proceedings of the ACL*.
- Strohman, T., D. Metzler, H. Turtle, W. B. Croft. 2005. INDRI: A Language-Model Based Search Engine for Complex Queries. *Proceedings of the International Conference on Intelligence Analysis*.
- Vanderwende, L., and A. Menezes, R. Snow. 2006. Microsoft Research at RTE-2: Syntactic Contributions in the Entailment Task: An Implementation. *Proceedings of the Second PASCAL Challenges Workshop on Recognising Textual Entailment*.
- Vorhees, E. 2008. Contradictions and Justifications: Extensions to the Textual Entailment Task. *Proceedings of the Meeting of the ACL:HLT*.
- Wang, C., M. Collins, and P. Koehn. 2007. Chinese Syntactic Reordering for Statistical Machine Translation. *Proceedings of the EMNLP and the Conference on Natural Language Learning*.
- Watanabe, T., E. Sumita, and H. Okuno. 2003. Chunk-Based Statistical Translation. *Proceedings of the ACL*. 303-310.
- White, J., D. Hunter, and J. Goldstein. 2008. Statistical Evaluation of Information Distillation Systems. *Proceedings The International Conference on Language Resources and Evaluation*.

- Yamada, K. and K. Knight. 2001. A Syntax Based Statistical Translation Model. *Proceedings of the ACL*. 523-530.
- Zelenko, D., C. Aone, and A. Richardella. 2003. Kernel Methods for Relation Extraction. *Journal of Machine Learning Research* 3: 1083-1106.
- Zens, R. and H. Ney. 2004. Improvements in Phrase Based Statistical Machine Translation. *Proceedings of the Meeting of the ACL:HLT*. 257-264.
- Zens, R., O. Bender, S. Hasan, S. Khadivi, E. Matusov, J. Xu, Y. Zhang, and H. Ney. 2005. The RWTH Phrase-Based Statistical Machine Translation System. *Proceedings of the IWSLT*. 155-162.
- Zheng, J., W. Wang, and N. Ayan. 2008. Development of SRI's Translation System for Broadcast News and Broadcast Conversations. *Proceedings of the ICSLP*.
- Zimmerman, M., D. Hakkani-Tür, J. Fung, N. Mirghafori, L. Gottlieb, E. Shriberg, and Y. Liu. 2006. The ICSI+ Multilingual Sentence Segmentation System. *Proceedings of the ICSLP*.

Chapter 5: Machine Translation Evaluation and Optimization

Editors: Bonnie Dorr, Joseph Olive, John McCary, and Caitlin Christianson

5.1 Introduction

Authors: Bonnie Dorr, Matt Snover, and Nitin Madnani

The evaluation of machine translation (MT) systems is a vital field of research, both for determining the effectiveness of existing MT systems and for optimizing the performance of MT systems. This part describes a range of different evaluation approaches used in the GALE community and introduces evaluation protocols and methodologies used in the program. We discuss the development and use of automatic, human, task-based and semi-automatic (human-in-the-loop) methods of evaluating machine translation, focusing on the use of a human-mediated translation error rate HTER as the evaluation standard used in GALE. We discuss the workflow associated with the use of this measure, including post editing, quality control, and scoring. We document the evaluation tasks, data, protocols, and results of recent GALE MT Evaluations. In addition, we present a range of different approaches for optimizing MT systems on the basis of different measures. We outline the requirements and specific problems when using different optimization approaches and describe how the characteristics of different MT metrics affect the optimization. Finally, we describe novel recent and ongoing work on the development of fully automatic MT evaluation metrics that have the potential to substantially improve the effectiveness of evaluation and optimization of MT systems.

Progress in the field of machine translation relies on assessing the quality of a new system through systematic evaluation, such that the new system can be shown to perform *better* than pre-existing systems. The difficulty arises in the definition of a *better* system. When assessing the quality of a translation, there is no single correct answer; rather, there may be any number of possible correct translations. In addition, when two translations are only partially correct – but in different ways – it is difficult to distinguish quality. Moreover, quality assessments may be dependent on the intended use for the translation, e.g., the tone of a translation may be crucial in some applications, but irrelevant in other applications.

Traditionally, there are two paradigms of machine translation evaluation: (1) *Glass Box* evaluation, which measures the quality of a system based upon internal system properties, and (2) *Black Box* evaluation, which measures the quality of a system based solely upon its output, without respect to the internal mechanisms of the translation system. Glass Box evaluation focuses upon an examination of the system's linguistic

coverage and the theories used to handle those linguistic phenomena. Individual linguistic components of the system may be examined and be subjected to black box evaluations. This method of evaluation was primarily focused on rule-based expert systems, rather than statistical systems.

Black Box evaluation, on the other hand, is concerned only with the objective behavior of the system upon a predetermined evaluation set. This method of evaluation is only a fair comparison of systems if the systems being tested were both designed to work on data that is of the same character as the evaluation set or, if not, the person testing the systems has the objective of testing robustness across different data types with variations in structure, genre, and style. This method has proved invaluable to the field of machine translation, enabling comparison of systems on the same test sets in order to determine whether a given change to a system is in fact an improvement. The method of actually measuring the performance of a system upon a test set is still a very active research area, and evaluation metrics of this type are the focus of this part.

Within black box approaches both *intrinsic* and *extrinsic* measures are used to assess the accuracy and usefulness of MT output. Intrinsic measures focus on the quality of MT output and often involve quality comparisons between MT output and a set of reference translations that are predetermined to be of high quality. Human intrinsic measures determine quality through human subjective judgments of certain characteristics of the output such as fluency and adequacy. Automatic intrinsic measures use an easily computed sentence similarity measure to produce rankings among MT systems by comparing the corresponding MT output against a fixed set of reference translations. The use of automatic metrics for system optimization of MT systems represents a significant breakthrough in the field of machine translation that has been used heavily in the GALE program. Eight automatic metrics are discussed in this part: BLEU, NIST, METEOR, and WER, PER, GTM, TER, and CDER. These measures have become an essential part of the machine translation research cycle, allowing rapid testing of new features and models, as well as providing a method for the automatic optimization of system parameters.

In contrast to intrinsic measures, extrinsic (task-based) measures are aimed to test the effectiveness of MT output with respect to a specific task. Two examples discussed in this part are extrinsic measures of human performance based on document exploitation task accuracy and measurements of human reading comprehension on machine translated texts. Extrinsic measures are aimed at testing the utility of machine translation. It is this utility testing aspect that served as a crucial foundation for justifying the establishment of the GALE program, wherein measures such as Defense Language Proficiency Test (DLPT) and HTER were first tested on a broad scale.

As HTER has become the primary evaluation criterion for the GALE MT program, a central component of evaluation research is that of relating HTER to automatic measures, with the motivation of establishing an automatic metric that can serve as a surrogate for HTER during system development. In addition, researchers in the GALE program have investigated different optimization techniques for tuning statistical MT systems and system combinations. (For a more elaborate description of system combinations, see the section on System Integration Framework in Chapter 6 of this book.) Researchers have also focused on the development of improved automatic evaluation metrics for machine

translation, with the goal of improving levels of correlation of metric scores with human judgments of translation quality.

The next three sections provide the definitional foundation for the remaining chapters. Section 5.1.1 focuses on the historical background for the approaches described in this part. Section 5.1.2 discusses evaluation metrics that rely upon human judgments, arguing for the continued use of human judgments in MT evaluations and describing the kinds of judgments that are commonly used. Common practices for generating such judgments are surveyed and efforts to improve the validity and repeatability of human judgments are discussed. Section 5.2.2 examines automatic evaluation metrics that do not require any human interaction (BLEU, NIST, METEOR, WER, PER, GTM, TER, and CDER) and also provides a brief description of the human-mediated standard used in GALE (HTER), relating it to automatic measures.

The following three sections turn to evaluation measures and approaches requiring human intervention. The primary goal of these new metrics is to improve the levels of correlation of metric scores with human judgments of translation quality, especially at the levels of documents and individual segments. 5.3 presents two human-in-the-loop extrinsic measures for evaluating the output of machine translation technology: human performance based on document exploitation task accuracy and measurements of human reading comprehension on machine translated texts. Both approaches seek to provide real MT users an alternative, accessible frame of reference for assessment of MT engines. 5.4 focuses on human post-editing, its challenges and its use in GALE. The entire post-editing process is detailed and the challenges of editor consistency are addressed. Section 5.4.4 provides a detailed overview of the GALE MT evaluations and their results, including a study comparing the different metrics.

The final two Chapters turn to techniques for using MT evaluation measures in system tuning, approaches to improving on existing measures, and approaches to designing innovative MT measures. 5.5 describes different optimization techniques (Simplex, Powel's method, etc.) used for tuning statistical MT systems and system combinations. Experiments that highlight different aspects of this optimization and different characteristics of MT metrics are discussed. Finally, 5.6 describes significant recent work by members of the GALE research community on developing improved automatic evaluation metrics for machine translation.

5.1.1. Historical Background

Authors: Bonnie Dorr, Mark Przybocki, Matt Snover, Audrey Le, Gregory Sanders, Sebastián Bronsart, Stephanie Strassel, Meghan Glenn

The task of evaluating machine translation quality, like machine translation itself, has a long history (Hutchins 2001). As far back as 1966, in Appendix 10 of ALPAC (1966), experiments were reported with human ratings of intelligibility, as well as the informativeness of a human translation seen after studying the machine translation. One of the legacies of DARPA MT evaluations in the early 1990's has been the use of human subjective judgments to "score" the semantic accuracy and fluency of MT outputs against one or more professionally produced human reference translations. The original hypothesis was that human subjects trained on these scoring tasks could serve as

adequately accurate stand ins for full-fledged professional translators trained to rate the MT outputs according to well-established standards.

Several methods of evaluation using human judgments are frequently employed in the machine translation community. In some cases, the quality of system output is measured directly, such as with human judgments; in other cases, it is measured by performing reading tests or other downstream tasks with the system output, and in still other cases it is measured by calculating the amount of work required to correct the system output.

Two of the most common human evaluation metrics are fluency and adequacy judgments (White 1994; Callison-Burch 2007). *Fluency* requires a speaker fluent in the target language to judge whether the system output is fluent, regardless of whether content of the output is an accurate translation of the source words.

Adequacy disregards the level of fluency in the system output and, as far as this is possible, measures whether the essential information in the source can be extracted from the system output. The requirements for an annotator of adequacy are stricter than for fluency, as the annotator must be bilingual in both the source and target language in order to judge whether the information is preserved across translation. In practice, an annotator fluent only in the target language could also annotate adequacy using a set of high quality human translations of the source sentence.

Fluency and adequacy are measured separately on each sentence in the system output and are usually judged on a five or seven point scale (Przybocki 2008). They are sometimes averaged to give a single numerical score to a system output. Some studies (Turian 2003; Snover 2006) have shown poor correlation between annotators using this method, bringing into question the reliability of this method. Nevertheless, human evaluation has been used as a baseline by which evaluation metrics are frequently judged. Judgments of semantic adequacy (and related ideas such as understandability or fluency) (Gates 1996; Nubel 1997) have continued to be employed as extremely useful benchmarks for the performance of MT systems and proposed MT metrics, even though the reliability of human judgments remains difficult (Turian 2003).

By the mid 1990's, the results of regular ARPA evaluations of MT led to doubts about the validity and reliability of human ratings of adequacy and fluency of MT output (King 1996) due to compounding factors such as the human evaluators' experience as translators/evaluators and the evaluators' familiarity with the system interfaces. The idea of using multiple *reference translations* for the first *fully automatic* measures was being seriously explored by 1990, when Niessen (2000) investigated a combination of Word Error Rate with sentence-by-sentence selection from multiple human references. Edit-distance metrics have since been explored as measures for MT quality (Frederking 1994; Knight 1994; King 1996).

Post-editing, where the system output is corrected after it is produced, is another common method of measuring translation quality, with more accurate translation requiring less editing and poor translations requiring large amounts of editing. This method suffers as an evaluation metric due to the large amount of work required by human annotators to correct system output, rather than quickly scoring it on an objective scale.

We will see in Section 5.2.3.1 that the notion of post-editing is still relevant in modern measures, e.g., “Human-mediated Translation Error Rate” HTER (Snover 2006), an approach to evaluating machine translation that was developed in and for the GALE program. Although HTER is built on human judgments, its greatest obvious weakness is that it is a purely quantitative metric that weights all errors equally, when in fact some edits, some translation errors, are of trivial importance while others such as some instances of polarity errors (*is* vs. *is not*) can be devastating. On the other hand, human judges are more likely to “forgive trivial” translation errors, or incorporate synonyms in their post edits. This would not be possible in an approach that uses Gold Standard references only.

Task oriented evaluation (White 2000; Jones 2006) is concerned with the effect of machine translation on downstream tasks that use the translation output as input. Recognizing named entities, document categorization, and the effect on reading speed and accuracy are all examples of task oriented evaluation that have been used to measure the quality of machine translation quality. The performance of human evaluators on these tasks is compared across system outputs, as well as to performance when using reference translations. This evaluation method is beneficial in that it measures how useful the system output is for a task rather than how close it is to the reference answers. Translations that are not judged to be perfect according to other measures may still prove to be fully adequate for downstream tasks. Similarly, translations of a certain type may achieve high scores by other measures, but may turn out to be completely inadequate for these tasks, e.g., because of the aforementioned polarity errors. In addition to tasks that require humans to process the system output, one could imagine using the system output in some automatic downstream process, such as information extraction. As an evaluation metric, such automated task evaluations are problematic as they compound errors in the translation process with errors in the downstream automated process, and at best can be used only to measure how useful that translation is for that particular implementation of that downstream automated task.

Automated metrics such as BLEU, NIST, METEOR, WER, PER, GTM, TER, CDER and now HTER (to be presented in Section 5.2.3.1) were developed due to the high costs, lack of repeatability, subjectivity, and slowness of evaluating machine translation output using human judgments, and the desire to enable automatic tuning of system parameters (Och 2003). These are based on automated comparisons between MT system outputs and human translations of the same source material, typically looking at n -gram matches. Coughlin (2003) examined the nature of the correlation between such n -gram based automated metrics and human judgments on a four-point acceptability scale of the same MT outputs, using 124 sentences from a range of European language pairs.

Automated metrics that can somehow give credit for paraphrases and/or synonymy have some intuitive merits. METEOR (Lavie 2004; Banerjee 2005) makes use of stemming and (at least for English) of synonymy apparent in WordNet synsets. TER and HTER allow some rearrangements, scoring a block move as a single edit, which may be particularly useful for target languages that have relatively free word order. Russo-Lassner (2005) explored the possible merits of exploiting paraphrase-like features in MT evaluations. As has been mentioned, not penalizing paraphrases is part of the motivation for using HTER in the GALE program.

Perhaps the closest competing approach to mechanical edit distance measures are those based on concepts of one sort or another. Sanders (2008) presented the evaluations used in the DARPA TRANSTAC program, which are based on purely quantitative scoring of the probability of a successful transfer of open-class content words. Other work based on concepts includes the use of Interlingua Interchange Format (Levin 2000), and predicate argument structures (Belvin 2004).

5.1.2. Human Subjective Judgments

Authors: Gregory Sanders, Mark Przybocki, Nitin Madnani, and Matthew Snover

Two central problems, measuring utility and/or measuring quality, face anyone who sets out to create a new or different evaluation of the outputs of MT systems. Either there is a clear idea of the specific purposes that the translation is to serve and the evaluation focuses on the utility (or acceptability) of the actual MT outputs, and/or there is some idea of what constitutes a better translation with the evaluation focused on the quality of the MT outputs.

The general notion of quality of a translation is, however, somewhat subjective, and the most accepted measure, the Gold Standard, so to speak, is to have truly bilingual human judges compare the source language inputs to the target language outputs and then somehow provide an opinion about the quality of the translation. Such judgments are most often provided on a multi-point scale.

MT systems that take live or recorded spoken inputs must include automatic speech recognition (ASR) abilities, also called speech-to-text (STT). If evaluation of ASR/STT is not desired, textual transcripts may be used as a surrogate input, constituting perfect ASR. Similarly, spoken outputs involve text-to-speech (TTS) abilities, and if evaluation of the TTS is not desired, the textual input to the TTS can be evaluated.

This section of the book focuses on subjective human judgments, whether from bilingual judges who know both the source and the target languages or from monolingual judges who know only the target language. We provide arguments for the continued use of human judgments in MT evaluations, as well as, contrasting approaches using automated metrics, non-subjective counts, or editing-based metrics. We explain what kinds of subjective human judgments are commonly used, survey several ways to generate such judgments, and discuss efforts to improve the validity and repeatability of those judgments.

One type of subjective human judgments involves bilingual human judges reading the source-language input and the corresponding target-language output (see Figure 5.1), then providing a rating on a multi-point scale. As can be seen in Figure 5.1, such ratings require a bilingual judge. The qualities to be rated are most commonly semantic adequacy (is the meaning correct), understandability (is the meaning easy to understand), and fluency (does the MT output have errors that no native speaker would produce). An example of a multi-point scale for semantic adequacy is shown in Figure 5.2. When judging fluency, the source-language inputs are not relevant, so ratings for fluency are usually obtained from monolingual judges.

What was said:

إي نعم إحنأ نصوي هالشغلات يومياً.

What it was translated into:

Yes indeed we do these things every

Figure 5.1: A sample source utterance and translation as presented to a bilingual judge.

Adequacy of translation

3 Completely adequate

2

1 Tending towards adequate

0

-1 Tending towards inadequate

-2

-3 Inadequate

Figure 5.2: Spoken Language Communication and Translation System for Tactical Use

A criticism of using bilingual judges is that their knowledge of the source language is not going to be available to monolingual users, which may hypothetically, lead the bilingual to give a score that is influenced by factors that are of no interest or importance to monolingual users of the translations. We are aware of no real evidence that this is a problem in practice.

We do note that the logistics of using bilingual judges for manual assessment of MT quality are significantly more difficult to arrange as compared to using monolingual judges.

5.1.2.1. Justification for Continued Use of Subjective Human Judgments

There are two strong arguments for using human judgments. The first is that translations are produced for human users, and human judgments are thus the right measure of the qualities of the translation. The second is that human understanding of the real world allows a human judge to assess the practical importance of errors in a translation. For example, if a source language sentence actually means, “There are new land-mines buried under the road to Baghdad,” and the machine translation is, “There are no land-mines buried under the road to Baghdad,” then the one small error (of polarity) is terribly important because lives can be at risk. A human judge should recognize and weigh the risk correctly.

Many properties make up the quality of a translation. *Semantic adequacy* (also known as fidelity) is often viewed as the most important, as it answers the question: does the translation have the same *meaning* as the source-language material?

Understandability is also important. A translation may have the completely correct meaning yet be so awkward as to be difficult to understand, noticeably more difficult than the source-language material. Or a source-language idiom may be translated literally and be impenetrable for a target-language hearer/reader. One cannot usefully ask a judge directly whether a translation is understandable: one must find a way to measure or test the judge's understanding.

Finally, fluency is important. The translation may have the correct meaning, be easy to understand, yet could be something that no native speaker of the target language would say or write, for example, "I talks Arabic real OK", or more severe fluency problems. When judging fluency, the source-language inputs are not relevant, so ratings for fluency are usually obtained from monolingual judges.

Although not discussed here, there are other properties of a translation that one might evaluate via subjective human judgments. For example, the target language output should be appropriate for the intended users. Examples of such problems include 'Britishisms' for American users, or vice-versa, female-specific forms when the hearer/reader is male and improper social register (generating language appropriate for speaking to a small child when the person addressed is a distinguished elder).

In the case of human translators using an MT system as an aid, the human translator's proficiency at using the computer interface tools and workstation environment is confounded with measurements of core MT performance (White 1994). This consideration may also be true of human judgments of machine translations, if the judges have difficulty understanding or using whatever system they must use to give their opinions.

Any evaluation of translation involving human judges is inherently also an evaluation of the human judges (King 1996). There are two main concerns with evaluations based on human judgments that must be addressed. The first is that humans will have different opinions and will give different answers for the same assessment. Generally a panel of independent judges is needed to average out those differences. Some of these differing opinions may be attributed to the unavoidable case that judges from different backgrounds tend to weight characteristics of a translation such as syntax "errors" or oddities of style differently (Nubel 1997). Some recent studies (Turian 2003; Snover 2006) have shown poor correlations between annotators using subjective judgments, further bringing into question the reliability of these types of human judgments. The second concern is that the evaluation will be affected by whether the judges are familiar with the subject matter and/or sub-language. One can imagine the types of assessment differences that might occur when using judges with no military experience to assess a translation between Marines that reads, "Tell me the identity of the adjacent unit" or using American judges to assess, "Drop that spanner! Pop the bonnet and boot!"

Nevertheless, human evaluation has been used as a baseline by which evaluation metrics are frequently judged. The fact is we have no real substitute for human judgments of translations. Such judgments constitute the reference notion of translation quality.

5.1.2.2. Survey of Common Practices

Several varied kinds of subjective human judgments of machine translation outputs have been tried. In this section, we discuss characteristics that could be used to describe approaches to subjective judgments, survey six approaches, and comment on their apparent problems and strengths. For purposes of giving clear examples of instructions to the judges, this section will assume the target language is English.

All approaches can be seen as a set of choices among a set of alternatives:

- Are the judges monolingual or bilingual?
- Do the judges have to understand the subject matter?
- Is there a reference for the judges? A reference could be either the source-language material or else one-or-more target language human translations.
- What characteristic(s) are to be measured: fluency, semantic adequacy, relative quality, acceptability?
- What type of judgment is to be made: a yes/no judgment (as of acceptability), a judgment on some sort of numeric scale (allowing correlation/regression analyses), a categorical judgment, or a preference judgment (possibly even a rank-ordering).
- How to analyze the judgments statistically or use them for formative or summative evaluation purposes.
- Is the goal to evaluate the translation of some particular text/audio item, or is the goal to evaluate the overall performance of the machine translation system? In practice, this particular question is about the use to be made of the evaluation results rather than the evaluation approach to be used.

<i>How do you judge the fluency of this translation?</i>	
<i>It is:</i>	
5	Flawless English
4	Good English
3	Non-native English
2	Disfluent English
1	Incomprehensible

Figure 5.3: One example of a multi-point fluency scale, with anchor text for each value

5.1.2.2.1. Fluency ratings from literate monolingual speakers of the target language.

For proper fluency assessments the judge should have access to only the translation being assessed, and not the original source data or a reference translation. The question put to the judges is typically along the lines of, “Is this good English?” It is common to ask the judge to choose a rating from a multi-point scale. Typically, the choices have some anchor value or text, as in Figure 5.3, which displays the scale (Linguistic Data Consortium 2005) that was used in annual NIST OpenMT evaluations and in a meta-evaluation (Callison-Burch, *et al.* 2007) that was done for the ACL-2007 Workshop on Statistical Machine Translation.

The most negative anchor text in the example above may be construed as conflating the notion of comprehensibility with that of fluency, an effect that could be lessened if the anchor text were instead “*Word salad / This is not English.*”

The main problem encountered with subjective human judgments of fluency is that understandability or comprehensibility of the translation enters into the picture. This is a problem if the judges are not familiar with the subject matter, and is a particularly acute problem if the source-language material is disfluent, as is often the case with spoken language, or unstructured texts such as web data.

This approach can be characterized with respect to our bullet list of choices. Fluency judgments can be given by monolingual judges and it is helpful if the judges understand the subject matter. Fluency judgments do not require a reference translation (or the source-language material). The quality to be judged is fluency. (Is this good English?) The judgments are usually given as categorical rankings, but the judgments are often analyzed as numeric – as seen in the numbers that begin the anchor values in our example.

Analyzing these judgments as categorical data is favored unless the judges are explicitly presented with the numeric interpretation of their choices and informed that those numbers are the data that will be analyzed and that the numeric values are presumed to be equally spaced.

5.1.2.2.2. Using fluency ratings from literate monolingual speakers of the target language as a proxy for accuracy ratings.

Wilks (2008) demonstrated that reasonably fluent MT output is also usually reasonably correct in meaning. For that reason, ratings of fluency by a monolingual judge often correlate with judgments of semantic adequacy sufficiently that (strange as the methodology might be) fluency judgments have some power to predict ratings of semantic adequacy.

In contrast, human language learners typically produce translations that are fluent even when the meaning is wrong, perhaps because a human translator (in contrast to a machine translation system) may construct a plausible meaning. A truly bilingual judge can detect such errors.

It is not known whether the correlation (between fluency and adequacy judgments of machine translation outputs) is a result of fluency judges assigning lower fluency scores when the MT output contradicts human knowledge of the real world.

With respect to the bullet list of choices, this approach is like the previous approach, but the caveats mentioned for this approach turn on the use of monolingual judges and the effects of the judges’ understanding of the subject matter. Note that although the judgments are of fluency, the quality to be measured, indirectly, is semantic adequacy. The judgments are usually given as categorical choices, but the judgments are often analyzed as numeric. It would probably be most appropriate to analyze these data as categorical and to attempt to validate this approach by showing some systematic relationship between the fluency judgments and some independent measure of semantic adequacy. The authors are not persuaded that this validity will always exist.

5.1.2.2.3. Semantic adequacy from truly bilingual judges.

This is widely regarded as the Gold Standard for assessment of MT quality. In addition to the translation being assessed, a bilingual judge will have access to the original source text and optionally a reference translation. The question put to the judge usually asks for a categorical rating based on the meaning contained in the translation as compared with the source text. It is necessary to use judges who understand the subject matter sufficiently for such judgments to be valid. In addition, for bilingual judges, their proficiency and native language is also important.

Typically, the human judgments are given on a multi-point scale, such as that in Figure 5.3.

A problem with using such a scale for any type of subjective human assessment is that the level of agreement between judges is typically not high, unless one counts the agreements as including the assessments that disagree by one category on a five-point scale, or by up to two categories when using a seven-point scale. This problem of inter-judge agreement can be dealt with by averaging over a panel of several judges.

One can treat the data as categorical and present the results for each judge separately, looking to see whether the judges agree about which items are better or worse. Or, if the scores have a valid numeric interpretation, then for comparisons where each judge gives scores on a set of many items, and all judges score the same set of items, one can normalize the data from each judge separately to a z-statistic (mean = 0.0 and standard deviation = 1.0).

However, if the goal is an absolute measurement of quality, or to compare performance between a current evaluation and a previous evaluation, normalizing across judges is of course not relevant. For comparisons to measure progress between performance on a current evaluation and performance on a previous evaluation, especially if the panel of judges changes from one evaluation to the next, one falls back on trying to hold the evaluation procedures as constant as possible – there is no statistical magic bullet in this situation. With a panel of several judges, more robust statistics such as a trimmed mean (say, counting only the middle five judges from a panel of nine), the median, or the inter-quartile range may be used.

If the goal is to assess fitness for some purpose, it may be possible to find a judge, or judges, who represent the intended users and whose assessments are known a-priori to accomplish that goal.

If a machine translation system or the translation itself is to be assessed, these judgments will be used directly. But, if an automated metric for machine translation quality must be assessed, that assessment is usually based on the degree of correlation between the scores from the automated metric and these ratings of semantic adequacy from bilingual judges.

Returning to our bullet list of characteristics of assessment techniques, these judgments are from truly bilingual judges and it is important that they understand the subject matter in order for their judgments to reflect the quality of the translation. Judgments of semantic adequacy require a reference translation (or the source-language material). The quality to be judged is semantic adequacy (i.e., whether the translation conveys the meaning). The judgments are usually given as categorical rankings, but may

be given as numeric, even as an effectively real-number value such as percentage. If the judgments are given on a categorical scale, the data should be analyzed as categorical unless the judges are presented with the numeric interpretation of their choices and informed that the judgments will be analyzed as those numbers.

5.1.2.2.4. Accuracy ratings from literate monolingual speakers of the target language who consult one or more careful human translations.

This is a widely used method for assessing the semantic adequacy of translations, due in part to the ease of obtaining monolingual judges. In addition to the translation being assessed, a monolingual judge will have access to one or more reference translations. The question put to the judge usually asks for a categorical rating based on the meaning contained in the translation as compared with the reference translation. An example is the following scale, Figure 5.4, in which “gold-standard” means a careful human reference translation.

<i>How much of the meaning expressed in the gold-standard translation is also expressed in the target translation?</i>	
5	All
4	Most
3	Much
2	Little
1	None

Figure 5.4: Another scale for semantic adequacy

In practice, this is popular because it tends to be more convenient and less expensive than the approach just discussed using bilingual judges. However, this approach has its limitations. When comparing a translation against a single human reference translation it relies on the reference translation as being perfect, a problem that is worse if the reference translation gives no alternatives for ambiguous passages. Everything mentioned elsewhere in this section about the need for the judges to understand the subject matter and sub-language applies equally to human translators. So, errors introduced by a single human translator will cascade through these subjective assessments.

Given somewhat difficult source-language material, and multiple human reference translations, one commonly finds that the human translators have disagreed about the meaning of the source material. Many such disagreements can only be understood by someone who understands the source language.

The balance switches, and criticisms evaporate, when one has several *completely independent* careful human translations, because they may bring out ambiguities that would otherwise have escaped the notice of a judge, or may fortuitously involve at least one translator who is sufficiently familiar with the subject matter as to understand the source material correctly. These advantages of multiple source language experts (the translators) independently giving careful consideration to the meaning of the source material can overcome all the edge of having bilingual judges who will work

independently (and thus not take advantage of insights to which only one judge arrives) and who may ponder the source material less thoroughly than careful translators due to focusing on only the errors of the machine translation. Of course, introducing several reference translations into the assessment process may increase the cognitive burden placed on the human judge.

With respect to the bullet list of characteristics of assessment techniques, these judgments are from monolingual judges, but in all other respects have the same characteristics as the version using bilingual judges.

5.1.2.2.5. Informativeness ratings by literate monolingual speakers of the target language.

The essence of this approach is to have the judge make a thorough attempt to understand the translation (perhaps by asking the judge to rewrite it so that it is completely clear), and then *afterwards* show the judge a careful human translation and have the judge rate how informative the reference human translation turned out to be. For example, “Did the reference translation change your understanding of what the passage meant?” This can be rated on a multi-point scale. This approach has been around since the earliest days of machine translation research: it was first described in the ALPAC report (ALPAC 1966). The crucial point of this approach is that the judge must thoroughly consider the translation and arrive at a frozen interpretation thereof before being allowed to see the reference translation; one wants to guard against the judge who realizes only in retrospect (after having seen the reference) that the meaning was obvious and then says the reference was not informative. Although one could simply ask the judge to arrive at a frozen understanding before viewing the reference, it is probably useful to force this issue. For example, one might ask the judge, before seeing the reference, to edit the translation to make it clear and fluent, an editing approach used by Callison-Burch *et al.* (2007).

The key advantage of obtaining ratings of the informativeness of a human translation is that it is a measure not just of semantic adequacy but also of how well a monolingual speaker of the target language understood the machine translation output before seeing a reference translation.

If judges are truly bilingual, a slightly different version of this evaluation can be performed, with the informativeness step showing the judge the source material (instead of a reference translation). However, this version gives up the advantage of having judges who are naive about the characteristics of the source language.

Returning to our bullet list of characteristics of assessment techniques is particularly interesting here because this approach is so different from those described so far. These judgments can be from either monolingual or bilingual judges. It is important for the judges to understand the subject matter in order for their judgments to reflect the quality of the translation. Judgments of informativeness require a reference human translation or translations. The quality to be judged is semantic adequacy (i.e., whether the translation conveys the meaning), but this quality is measured in an interestingly indirect way that encompasses the understandability of the translation. Although the judgments could be given as a yes/no response, it is far more useful to obtain responses where the categories

get at the degree of change in the judges understanding and/or ask the judge whether the problem was with the meaning of the MT output or it's comprehensibility.

5.1.2.2.6. Preference ratings by a literate monolingual speaker of the target language or from bilingual judges.

Often used as a quick approximation for improvement in the development work of system building, this approach presents a monolingual judge with a pair of translations and the corresponding reference translation or presents a bilingual judge with the pair of translations plus the source material. The question put to the judge is along the lines "Which translation do you prefer?"

If one wishes to rank-order multiple translations of the same material, one may use pair-wise preference judgments to build a rank-ordering across several systems. Alternatively, one can present all the versions together and ask the judge to rank-order them.

For a formative evaluation, one might wish to ask the judges to indicate why they prefer one translation over the other, although the judge may not be able to provide an accurate answer.

Our bullet list of characteristics of assessment techniques is also interesting here. These judgments can be from monolingual or bilingual judges. It is important for the judges to understand the subject matter. Monolingual judges require one or more high quality reference human translations and bilingual judges should be provided with the source-language material (either instead of, or in addition to, the human reference translations). The quality to be judged should be overall quality – it is unreasonable to believe judges could isolate quality such as semantic adequacy when making preference judgments. The judgments are preferences which could be implemented in a full pair-wise fashion, comparing two system translations at a time, or in theory, judges could be presented with several translations and asked to rank order their preferences. Rank-ordering may be more burdensome and time-consuming but would avoid the $A > B$, $B > C$, and $C > A$ scenario that one might find with pair-wise judgments. There is reason to believe pair-wise rankings will be performed more reliability than directly rank ordering several translations. The data should be analyzed as ranks.

5.2 Automatic and Semi-Automatic Measures

5.2.1. Automatic and Semi-Automatic Measures: Introduction

Authors: Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz

Automatic machine translation evaluation metrics were developed due to the high costs, lack of repeatability, subjectivity, and slowness of evaluating machine translation output using human judgments, and the desire to enable automatic tuning of system

parameters (Och 2003). These automatic measures generally judge the quality of machine translation output by comparing the system output, also referred to as candidates or hypotheses, against a set of reference translations of the source data.

The quality of these evaluation metrics is usually measured by determining the correlation of the scores assigned by the evaluation metrics to scores assigned by a human evaluation metric, most commonly fluency and adequacy. In some cases, single scores for systems averaged over entire test sets are compared, and in others document or even sentence level scores are compared. The use of Spearman versus Pearson correlation coefficients also varies within the field. An alternate, although rarely examined, method of comparing automatic evaluation metrics is to examine their effect upon parameter tuning of the MT system. It has not been established that an MT system, which has its parameters tuned to maximize the performance on a metric that correlates better with human judgments, will actually produce better MT output. Such a study is hampered by both the varying parameters of MT systems which allow different aspects of different MT systems to be controlled in different ways as well as variance in parameter optimization methods. Such a study is necessary though for a full understanding of the automatic evaluation of machine translation.

This section details some of the more common automatic evaluation metrics that are used for MT evaluation and parameter optimization, as well as one semiautomatic variant. Word Error Rate (WER) (Section 5.2.2.1) the standard metric of Automatic Speech Recognition performance and one of the first automatic metrics applied to machine translation. BLEU (Section 5.2.2.2), a widely used precision oriented measure that counts the n -gram matches between the MT output and the Gold Standard references. (NIST is a variant of BLEU that is weighted according to n -gram informativeness.) METEOR (Section 5.2.2.3), an alignment-based measure that utilizes stemming and synonymy to better align the MT output with the Gold Standard references. Translation Edit Rate (TER) (Section 5.2.2.4), an extension of the Word Error Rate measure WER (and its variants, MWR, PER and GTM) that allows the movement of word sequences within the MT output. All of these automatic measures rely upon a comparison of the MT output to a set of Gold Standard references, which represent only a small sample of the set of possible correct translations. Human-mediated TER (HTER) (Section 5.2.3.1), which is used as the primary criterion for evaluating the MT systems in GALE, addresses this lack of Gold-Standard references by having human annotators create new references that are as similar as possible to the MT system output, and evaluating the system output with TER using these targeted references.

These automatic metrics represent just a small sampling of the automatic evaluation metrics that been proposed in the machine translation research community and were chosen due to their prominence and use in the GALE research community.

5.2.2. Automatic Measures

Authors: Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz

5.2.2.1. Word Error Rate (WER MWER) and Position-Independent Error Rate.

One of the first automatic metrics used to evaluate MT systems was Word Error Rate (WER), which is the standard evaluation metric for Automatic Speech Recognition. WER is computed as the Levenshtein distance (Levenshtein 1966) between the words of the system output and the words of the reference translation divided by the length of the reference translation. The Levenshtein distance is computed using dynamic programming to find the optimal alignment between the MT output and the reference translation, with each word in the MT output aligning to either 1 or 0 words in the reference translation, and vice versa. Those cases where a reference word is aligned to nothing are labeled as *deletions*, whereas the alignment of a word from the MT output to nothing is an *insertion*. If a reference word matches the MT output word it is aligned to, this is marked as a *match*, and otherwise is a *substitution*. The WER is then the sums of the number of substitutions (S), insertions (I), and deletions (D) divided by the number of words in the reference translation (N) as shown in Equation (5.1).

$$WER = \frac{S+I+D}{N} \quad (5.1)$$

MWER (Multi-Reference WER) (Nießen 2000) – the application of WER to more than one reference translation – refers to the minimum of the WER scores between the MT output and each reference. In essence, MWER is the WER between the MT output and the closest reference translation. While this allows WER to be used with multiple references, the references are not combined in any fashion and are not truly exploited by the metric.

Unlike speech recognition, there are many correct translations for any given foreign sentence. These correct translations differ not only in their word choice but also in the order in which the words occur. WER is generally seen as inadequate for evaluation for machine translation as it fails to adequately combine knowledge from multiple reference translations and also fails to model the reordering of words and phrases in translation.

Position-independent Error Rate or (PER) (Tillmann 1997) is an attempt to address the word-ordering limitation of WER by treating the reference and hypothesis as bags of words, so that words from the hypothesis can be aligned to words in the reference regardless of position. Because of this the PER of an MT output is guaranteed to be lower than or equal to the WER of the MT output. This variant has the disadvantage of being unable to distinguish a correct translation from one where the words have been scrambled.

5.2.2.2. BLEU

BLEU (Bilingual Evaluation Understudy) (Papineni 2002) is the current standard for automatic machine translation evaluation. Like MWER, a key characteristics of BLEU is its direct exploitation of multiple references. The BLEU score of a system output is

calculated by counting the number of n -grams, or word sequences, in the system output that occur in the set of reference translations.¹ BLEU is a precision-oriented metric in that it measures how much of the system output is correct, rather than measuring whether the references are fully reproduced in the system output. BLEU could be gamed by producing very short system outputs consisting only of highly confident n -grams, if it were not for the use of a brevity penalty which penalizes the BLEU score if the system output is shorter than the references.

$$p_n = \frac{\sum_{c \in \{can\}} \sum_{n\text{-gram} \in c} \text{Cnt}_{clip}(n\text{-gram})}{\sum_{c' \in \{can\}} \sum_{n\text{-gram}' \in c'} \text{Cnt}(n\text{-gram}')} \quad (5.2)$$

$$BP = \begin{cases} 1, & \text{if } c > r; \\ e^{(1-\frac{r}{c})}, & \text{if } c < r. \end{cases} \quad (5.3)$$

$$BLEU = BP \cdot \exp(\sum_{n=1}^N w_n \log p_n) \quad (5.4)$$

Equation (5.2) shows the computation of the BLEU precision scores for n -grams of length n , where Can are the sentences in the test-corpus, $\text{Cnt}(n\text{-gram})$ is the number of times an n -gram occurs in a candidate, and $\text{Cnt}_{clip}(n\text{-gram})$ is the minimum of the unclipped count and the maximum number of times it occurs in a reference translation. Equation (5.3) shows the calculation of the BLEU brevity penalty, where c is the length of the candidate translation, and r is the length of the reference translation. These terms are combined, as shown in Equation (5.4), to calculate the total BLEU score, where N is typically 4, and w_n is usually set to $1/N$.

Several variants of the BLEU measure are commonly used that primarily differ in the calculation of the number of reference words used to calculate the brevity penalty. The IBM version of BLEU uses the average lengths of the references, while the NIST version of BLEU uses the shortest reference to calculate brevity penalty. The NIST automatic evaluation metric, not to be confused with the NIST version of BLEU, is a variant of the BLEU metric that differs in that it uses the arithmetic mean of the n -gram counts, rather than the geometric mean used in BLEU.² More importantly, the NIST metric does not treat all n -grams equally, but weights them according to their informativeness. The frequency of an n -gram sequence is calculated from the set of reference translations, rather than from an external corpus.

¹ A maximum length of four words is common.

² <http://www.itl.nist.gov/iad/mig/tests/mt/doc/ngram-study.pdf>

Since its introduction, BLEU has become widespread in the machine translation community and is the most commonly reported evaluation metric. Several shortcomings of the BLEU evaluation metric have been brought forth by the measure's critics (Turian 2003; Lavie 2004; Callison-Burch 2006). One of the primary critiques of BLEU is absence of recall in its formulations. In addition, BLEU was designed for, and has been shown to work best when used on, large test corpora, such that the scores are averaged over many sentences. BLEU scores of individual sentences are not considered reliable. Other shortcomings of BLEU are the lack of synonym matching and the inability to detect multiple proper word orders. In short, translation quality – especially semantic quality – is not detected by the BLEU measure.

A number of new automatic evaluation measures for machine translation have been proposed in recent years to compensate for the perceived failings of the BLEU scoring measure. These measures all fundamentally deal with the notion of string matching between reference translations and hypothesized translations. The following sections describe several of these more recent measures in detail.

Despite these criticisms, BLEU remains the most commonly used automatic metric both for the optimization of system parameters and for final evaluation of the quality of an MT system. The use of the BLEU metric has driven development in the MT research community, and it is now the automatic evaluation metric against which all new metrics are compared.

5.2.2.3. METEOR

METEOR (Metric for Evaluation of Translation with Explicit ORdering) (Banerjee 2005) is an evaluation specifically designed to address several observed weaknesses in BLEU. METEOR is a tunable, alignment-oriented metric, whereas BLEU is generally a precision-oriented metric.³ Unlike BLEU, which only calculates precision, METEOR calculates both precision and recall, and combines the two, as shown in Equation (5.5), to calculate the parameterized harmonic mean.⁴ In more recent work (Lavie 2007), higher correlations with human judgments were obtained by optimizing the parameters of the harmonic mean for specific target languages.

$$F_{mean} = \frac{P \cdot R}{\alpha P + (1 - \alpha) R} \quad (5.5)$$

METEOR uses several stages of word matching between the system output and the reference translations in order to align the two strings. The matching stages are as follows:

³ The brevity penalty in BLEU addresses this issue by penalizing short translations, which BLEU would otherwise be unfairly biased towards. Without the brevity penalty, BLEU would be purely a precision-oriented metric.

⁴ The default parameters for the harmonic mean originally set $\alpha = 0.9$. It is because of this strong bias towards recall that METEOR is often considered to be a recall-oriented metric.

1. **Exact matching.** Strings which are identical in the reference and the hypothesis are aligned.
2. **Stem matching.** Stemming is performed, so that words with the same morphological root are aligned.
3. **Synonymy matching.** Words which are synonyms according to WordNet (Fellbaum 1998) are aligned.

In each of these stages, only words that were not matched in previous stages are allowed to be matched. Only unigrams, single words, are compared for matches. Precision in METEOR is defined as number of matches divided by the number of words in the system output, and recall is defined as the number of matches divided by the number of words in the reference.

In addition to the F_{mean} , METEOR also uses a fragmentation penalty to penalize system outputs composed of many small chunks, where a chunk is defined as a sequence of matches that are contiguous and identically ordered in the system output and reference. Fragmentation is calculated as the number of chunks divided by the number of unigram matches. The fragmentation is calculated as shown in Equation (5.6), with default parameters of $\beta = 3.0$ and $\gamma = .5$.

$$Pen = \gamma \cdot frag^{\beta} \quad (5.6)$$

This fragmentation penalty causes METEOR to correctly penalize “word salad” MT output that would be allowed under the PER metric, and is an essential portion of the METEOR scoring metric.

The final METEOR score is calculated as: $score = (1 - Pen) \cdot F_{mean}$.

The free parameters (alpha, beta, and gamma) were originally hand-set to alpha=0.9, beta=3.0, and gamma=0.5, encoding a strong preference for recall and strict fragmentation penalty. In current versions of METEOR, these parameters are tuned to maximize correlation with human judgments of translation quality. Sections 5.6.2.2 and 5.6.2.3 describe tuned parameters and tuning procedures. These tuned parameter values are now the defaults.

Unlike BLEU, METEOR does not penalize longer answers and incorporates a level of linguistic knowledge in the form of its stem and synonym matching allowing it to identify equivalences between the MT output and the reference translation that would be ignored by these earlier measures. METEOR lacks one of BLEU’s key features however: the direct exploitation of multiple references, as METEOR cannot combine knowledge from multiple references into its score, but rather, when multiple references are available, METEOR selects the reference translation for each segment that gives the best METEOR score. Furthermore, METEOR’S ability to handle variability via stemming and synonyms already reduces the expected gain from comparing against multiple references simultaneously, mitigating the effect of this limitation.

The highly recall based measure though can be exploited by the inclusion of additional highly likely words (such as “the” in English) in the MT output, giving higher scores to outputs with these additional padded words – although such behavior is not

typically exhibited by modern machine translation systems. The benefits of a precision-oriented metric such as BLEU versus a recall-oriented metric such as METEOR must be tested experimentally in terms of correlation with human judgment.

5.2.2.4. General Text Matcher (GTM), Translation Edit Rate (TER) and CDER

As discussed in Section 5.2.2.1, WER (in its standard form) is viewed as inadequate for machine translation evaluation due to its failure to utilize multiple reference translations and its failure to model the reordering of words and phrases in a translation. This section discusses several recent evaluation metrics that attempt to address this latter failing.

The General Text Matcher (GTM) (Turian 2003) evaluation metric attempts to model the movement of phrases during translation by using the maximum matching size to compute the quality of a translation. It finds the longest sequences of words that match between the hypothesis and the reference.⁵ The size of the matches is defined in Equation (5.7), where M is the set of matches found. This formulation can be generalized to other exponents, and in experimental results (Turian 2003) the metric appears to work best when $size(M) = \sum_{r \in M} length(r)$. The precision and recall are computed as the size of the matches divided by the length of the system output or the reference, respectively. The score of the GTM is the harmonic mean, F-score, of precision and recall.

$$size(M) = \sqrt{\sum_{r \in M} length(r)^2} \quad (5.7)$$

GTM can also be used with multiple references by concatenating the references together, and not allowing a match to cross the boundary between references.

Like BLEU (and also TER, discussed below), GTM does not incorporate any linguistic knowledge, and only considers words in the MT output and the references as matching if they are identical. The typical definition $size(M)$ also generates several interesting properties that are not present in TER however. If a single word in the translation is incorrect, the affect on TER is the addition of a single edit, while in GTM the effect depends upon the location of the error in the sentence. An error in the middle of the sentence has the largest effect, resulting in two matches of equal size, while an error closer to either end of the sentence has a minimal effect, only reducing the length of the match by 1 if the error is the first or last word. Whether errors in the middle of the sentence are of much greater importance than errors at the beginning or end of the sentence, rather than such errors being of equal importance, as in TER, has not been adequately studied.

⁵ Because the solution to this is conjectured to be NP-hard, a greedy search is used to iteratively select the longest sequences. The authors claim that this obtains the optimal solution in 99% of cases.

Translation Error Rate (TER) (Snover 2006) addresses the phrase reordering failing of WER by allowing block movement of words, also called *shifts*, within the hypothesis as a low cost edit, a cost of 1, the same as the cost for inserting, deleting or substituting a word. While a general solution to WER with block movement is NP-Complete (Lopresti,1997), TER addresses this by using a greedy search to select the words to be shifted, and as well as further constraints on the words to be shifted. These constraints are intended to simulate the way in which a human editor might choose the words to shift.

The shifting constraints used by TER serve to both reduce the computational complexity of the model and better model the quality of translation. Examining a larger set of shifts, or choosing them in a more optimal fashion might result in a lower TER score, but it would not necessarily improve the ability of the measure to determine the quality of a translation. The constraints used by TER are as follows:

1. Shifts are selected by a greedy algorithm that selects the shift that most reduces the WER between the reference and the system output.
2. The sequence of words shifted in the system output must exactly match the sequence of words in the reference that it is being shifted to.
3. The words to be shifted must contain at least one error, according to the WER, before being shifted. This prevents the shifting of words that currently correctly matched.
4. The matching words in reference that are being shifted to must also contain at least one error. This prevents shifting to align to words that already correctly aligned.

When TER is used in the case of multiple references, it does not combine the references, but scores the hypothesis against each reference individually. The reference with which the hypothesis has the fewest number of edits is deemed the closest reference, and that number of edits is used as the numerator for calculating the TER score, as is done in MWER. Rather than use the number of the words in the closest reference as the denominator, TER uses the average number of words across all of the references. Thus, the equation for the TER score, where *SUB*, *INS*, *DEL* and *SHIFT* are the number of substitutions, insertions, deletions and shifts, respectively, and \bar{N} is the average number of reference words, is shown in the following equation.

$$TER = \frac{SUB+INS+DEL+SHIFT}{\bar{N}} \quad (5.8)$$

TER cannot exploit multiple references as is done in BLEU nor does it incorporate external linguistic knowledge, e.g., synonyms, as is done in METEOR. These failings can be addressed through the use of targeted references – references created by humans specifically for a particular machine translation output – changing it from an automatic metric (TER) to a semi-automatic metric (HTER) as is discussed in Section 5.2.3.

Other automatic metrics for MT evaluation exist that follow the general formulation as TER, but address the complexity of shifting in different ways, such as CDER (Leusch 2006) (Cover Disjoint Error Rate). CDER exploits the fact that the number of blocks in a sentence is equal to the number of gaps among the blocks plus one. Thus, the block movements can equivalently be expressed as *long jump* operations that jump over the gaps between two blocks. The costs of a long jump are considered constant. The blocks are read in the order of one of the sentences. These long jumps are combined with the “classical” Levenshtein edit operations, namely *insertion*, *deletion*, *substitution*, and the zero-cost operation *identity*. The resulting long jump distance d_{LJ} gives the minimum number of operations which are necessary to transform the candidate sentence into the reference sentence. Like the Levenshtein distance, the long jump distance can be depicted using an alignment grid as shown in

Figure 5.5. Here, each grid point corresponds to a pair of inter-word positions in candidate and reference sentence, respectively. d_{LJ} is the minimum cost of a path between the lower left (first) and the upper right (last) alignment grid point which covers all reference and candidate words. Deletions and insertions correspond to horizontal and vertical edges, respectively. Substitutions and identity operations correspond to diagonal edges. Edges between arbitrary grid points from the same row correspond to long jump operations. It is easy to see that d_{LJ} is symmetrical.

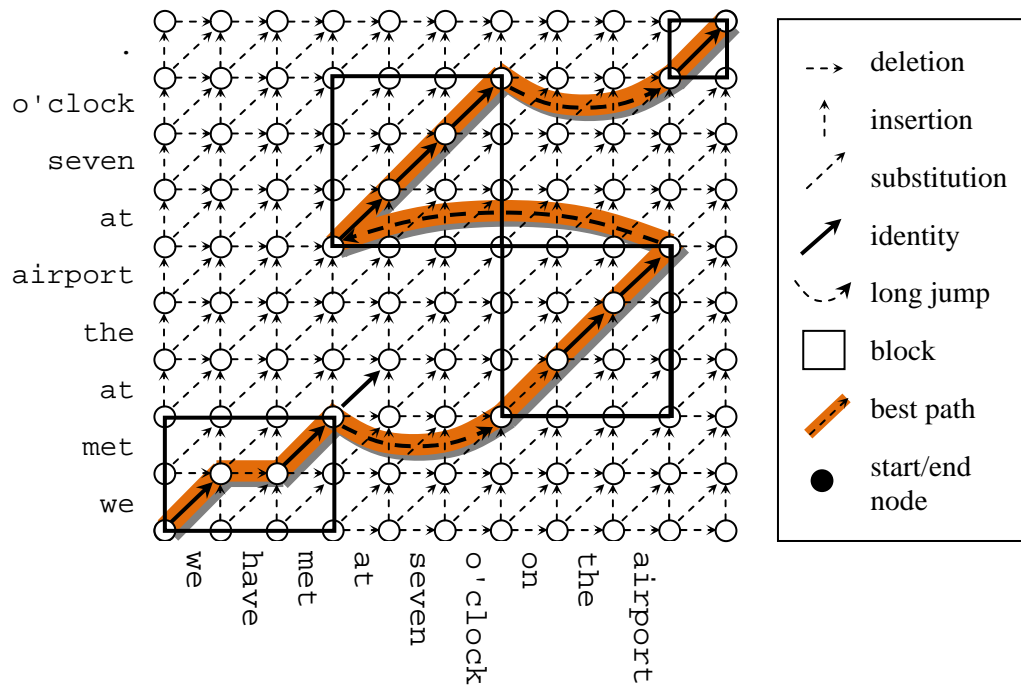


Figure 5.5: Example of a long jump alignment grid. All possible deletion, insertion, identity and substitution operations are depicted. Only long jump edges from the best path are drawn.

While finding an optimal path in a long jump alignment grid is an NP-hard problem, CDER uses an approach which has a suitable run-time, while still maintaining completeness of the calculated measure. The idea of the proposed method is to drop some restrictions on the alignment path.

The long jump distance, as well as the Levenshtein distance, require both reference and candidate translation to be covered *completely* and *disjointly*. When extending the metric by block movements, we drop this constraint for the candidate translation. That is, only the words in the reference sentence have to be covered exactly once, whereas those in the candidate sentence can be covered zero, one, or multiple times. Dropping the constraints allows for an efficient computation of the distance. CDER drops the constraints for the candidate sentence and not for the reference sentence to prevent any information contained in the reference from being omitted. Moreover, the reference translation will not contain unnecessary repetitions of blocks.

CDER can thus be seen as a measure oriented towards *recall*, while measures like BLEU are guided by *precision*. The CDER is based on the $\overline{CD}CD$ distance introduced by Lopresti and Tomkins (Lopresti 1997).⁶ It can be shown that the problem of finding the optimal solution can be solved in $O(I^2 \cdot J)$ time, where I is the length of the candidate sentence and J the length of the reference sentence. Using a modification of the Levenshtein algorithm, the time complexity can be further reduced to $O(I \cdot J)$ (Leusch 2006).

5.2.3. Semi-automatic Measures

Authors: Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz

5.2.3.1. HTER

HTER (Human-mediated Translation Error Rate) (Snover 2006) is a human-in-the-loop variant of TER that has also been used to evaluate machine translation systems. HTER requires the use of mono-lingual human annotators who create references that are targeted to a particular system output. Targeted references are crafted by changing the system output with a minimal number of edits so that it is both fluent and preserves the meaning of the other reference translations. Because a minimal number of edits are used to correct the system output, creating a targeted reference can be thought of as selecting from the set of all possible references the one which is closest, as measured by TER, to the system output. Targeted references could be used with any other automatic evaluation metric that uses reference translations. The use of targeted references has been shown to increase the correlation of automatic metrics with subjective human judgments, and can be seen as a method of addressing the sparsity of reference translations.

Because targeted references are tailored for each particular system output, they cannot be reused for system output from different systems or even output from a different version of the original machine translation system. Because targeted references cannot be

⁶ C stands for *cover* and D for *disjoint*.

reused and the need for human annotators to create targeted references, HTER is completely unsuited as a purely automatic machine translation evaluation metric. Viewed as a type of human judgment though, HTER is an expensive, but fine grained measure of translation quality, and has been used as the primary metric of MT performance in the GALE program. The full details of HTER as implemented in the GALE program are discussed in Section 5.4.2.

Although HTER is built on human judgments, its most obvious weakness is that it is a purely quantitative metric that weights all errors equally, when in fact some edits, some translation errors, are of trivial importance while others such as some instances of polarity errors (is vs. is not) can be devastating. Such a problem could possibly be addressed by using an automatic measure other than TER to compute the resulting error between the targeted reference and the hypothesis, although this leaves the question of which errors to more heavily weight. The answer to this is likely to depend heavily on the intended use of the translation.⁷

5.3 Tasks and Human-in-the-Loop Measures

5.3.1. Tasks and Human-in-the-Loop Measures: Introduction

Authors: Clare Voss and Doug Jones

In addition to the use of human judgments for comparison-based approaches to MT evaluation, humans are also central to extrinsic measures that have been adopted for the purpose of evaluating the output of machine translation. Two examples are measurements of human performance based on document-exploitation task accuracy and measurements of human reading comprehension on MT texts. Both approaches seek to provide real MT users with an alternative, accessible frame of reference for assessment of MT engines. Since machine translation may distort or delete portions of the original content, task developers in both approaches must distinguish assessments of:

- Mistakes in the MT text proper, that can be measured indirectly by human subjective judgments, as described in Section 5.1.2 or directly by intrinsic metrics with automated comparisons to human reference translations, as described in Section 5.2.2.
- Human response errors – whether caused by inherent difficulty of the text-handling task, human performance factors (such as fatigue, inadequate task training, or other individual differences), or mistakes in the MT text.

Developers of task metrics must: *first* establish the ground truth or answer set for the task and document collection, which entails understanding its foreign language content and typically building reference translations of the collection; *second*, after developing

⁷ We note that, in recent variations of HTER, the evaluator is able to specify the “weight” of different types of edits, thus enabling a task-oriented evaluation approach, where certain errors may be penalized more heavily than others.

the task protocol and establishing the experimental design and statistics, train the subjects and conduct the task under experimental conditions with MT documents; and *third*, assess the accuracy of task responses in relation to information conveyed in the test documents.

5.3.2. Document-Exploitation Task Accuracy

Authors: Clare Voss and Calandra Tate Moore

5.3.2.1. Introduction

Document collectors in the field, analysts at tactical operational centers, and translators operating from remote sites must work rapidly and accurately in triaging foreign language documents so that they or others in the shared workflow can prioritize which documents require additional analysis or *document exploitation* (DocEx). This section describes the experimental challenges and research results in assessing the effectiveness of Arabic-English MT engines for one document exploitation task, in enabling English speakers to identify essential elements of information in machine-translated documents that they otherwise would not understand.

5.3.2.2. Document Exploitation

When foreign language documents are collected by English-speaking troops in the field, the time-critical challenge is to determine the mission relevance of the information contained in those documents. In this context, “documents” refer to any of a wide variety of items, including hard-copy materials such as newspapers, leaflets, official reports, receipts, supply lists or pocket litter, as well as soft-copy files with text, images, video, audio, or any combination of mixed media.

When documents are printed, handwritten, or spoken in languages other than English, the DocEx process will at some point include “content translation” in conjunction with other text-handling tasks that range from simple filtering or topic detection to binning or triaging by prioritized categories, up to more linguistically complex tasks of entity extraction, event identification, and summarization within and across documents (see Table 5.1). The most severe bottlenecks in the DocEx process in the field occur when massive quantities of foreign language documents need to be reviewed, and too few translators are available for this work.

Task	Description of Task
Publishing	Produce a technically correct document in fluent English
Gisting	Produce a summary of the document in English
Extraction	For documents of interest, capture specified key information in English: <ul style="list-style-type: none"> • <i>Deep Extraction</i>: Event identification (scenarios): id incident type & facts <ul style="list-style-type: none"> – task iii: Event Completion: id relations among elements, time, & place of event • <i>Intermediate Extraction</i>: Relationship id (ex. member-of, phone-number-for) <ul style="list-style-type: none"> – task ii: Wh-item extraction: id who-, where-, when-type elements of information • <i>Shallow Extraction</i>: Named entity recognition: isolate names of people, places, organizations, times
Triage	For documents determined to be of interest, rank by importance <ul style="list-style-type: none"> – task i: Topic id: bin document by topic
Detection	Find documents of interest
Filtering	Discard irrelevant documents

Table 5.1: Hierarchy of text-handling tasks proposed by Taylor & White (1998), augmented to include ARL’s tasks i—iii

5.3.2.3. DocEx Tasks with MT

Shortly after Church (1993) described “good applications for crummy MT,” ARL⁸ decided to test a DocEx triage task as an application for FALCon,⁹ its laptop-based system for non-translators to scan in hardcopy documents, OCR the stored images to text, run the text through MT to convert it into “English”, and then search the resulting text for English keywords (Fisher 1997; Fisher 1999). The goal was for document collectors in the field to convert found documents into “English” via FALCon, and then to triage the output text for mission relevance, reducing the translators’ workload.

The FALCon systems, once in the field, however, came into the hands of in-country translators; they valued the systems for quickly generating first-pass translations that they could readily revise and give to their English-speaking supervisor, who in turn would review and possibly post-edit their translations for inclusion in summary reports. Instead of *reducing the number of documents* in the translators’ workflow by enabling others to triage documents, FALCon’s utility came in *reducing the time-to-translate documents* in the translators’ workflow by speeding up their process of gisting documents. With evidence coming back to the lab for the utility of FALCon in the field, two types of evaluation questions were raised about providing support for these systems:

- *Developmental testing (DT)* conducted by the developers in the lab: how “good” are the MT engines? (measures of translation performance)
- *Operational testing (OT)* conducted with potential users on working systems: how “effective” are users of the MT engines? (measures of task effectiveness)

In 2000, we began conducting pilot studies to determine which MT engines conveyed sufficient information in their translations to support different levels of text-handling tasks. In 2004, we ran two large-scale experiments on Topic Binning and Wh-item

⁸ Army Research Laboratory

⁹ FALCon stands for **F**orward **A**rea **L**anguage **C**onverter

Extraction and one large-scale pilot test on Event Completion (see tasks i-iii respectively, in Table 5.1) to assess three types of Arabic-English MT engines: MT-1 a rule-based machine translation (RBMT) engine, MT-2 a statistical machine translation (SMT) engine, and MT-3 a lexicon-based machine translation (LBMT) engine.¹⁰ This section focuses on operational testing (OT) of MT engines for Wh-item Extraction (task ii), where the engines showed statistically significant differences on two response metrics. The section concludes with analyses of how closely these task-based (OT) results align with text-based (DT) automated metric scores on the same set of documents.

Who-type: people, roles, organizations, companies, groups of people, and the government of a country
When-type: dates, times, duration or frequency in time, including proper names for days and common nouns referring to time periods
Where-type: geographic regions, facilities, buildings, landmarks, spatial relations, distances, and paths

Table 5.2: Wh-types in Wh-item extraction (task ii)

5.3.2.4. Wh-item Extraction Task

On this task, the subjects were asked to read through a machine-translated text on their screen and “mark up” all word sequences that belonged to the assigned wh-type for that text (see Table 5.2). Each marked-up sequence appeared both highlighted directly in the text in the wh-type color code and copied as an entry in the screen’s summary box with all wh-items extracted so far for the presented text.¹¹

The Arabic ground truth (GT) of wh-items was defined in the original Arabic documents. Then a corresponding English reference translation (RT) set of wh-items was derived with the translator of the task documents. Since subjects were asked to mark wh-items in the MT output texts, a corresponding set of correct answers for scoring their responses also had to be identified in these texts. Two task developers and one adjudicator coded the answer set of wh-items by marking up each MT text alongside its corresponding RT text already marked with RT wh-items. Their answer set was called the *omniscient truth* set of wh-items, to reflect their use of the reference translations while marking the MT texts.¹² The task developers assigned a category code to each wh-item for computing inter-coder reliability and reconciling and recoding the texts. The codes of A (correct), B (almost correct), and S (split) are defined in Table 5.3. When no words in the MT output were clearly semantically or phonetically related to an RT wh-item, that item was coded Z for lost in translation. As an example, three different codings

¹⁰ Results were presented at TIDES PI meetings (2001, 2002) and NIST Open MT Workshops (2003, 2004). DARPA TIDES funded the initial task-based evaluation research; the Center for Advanced Study of Language at U. of Maryland funded the follow-on large-scale experiments. Tasks i and iii yielded no statistically significant differences in the accuracy of subjects’ responses across MT engines: all MTs supported task i and no MTs supported task iii. (Laoudi 2006)

¹¹ For further details, see Tate and Voss 2006 and Voss and Tate 2006

¹² Note that the HTER post-editors are also “omniscient” in this sense: they read RTs to understand the SL text so that they can post-edit the MT output to convey its meaning exactly.

for the same underlying wh-item with different MT output results are shown in Figure 5.6.

A 1) Exact match, synonym, or paraphrase, where words are in grammatical word order 2) Contiguous phrase
B 1) Exact match, synonym, paraphrase, but wording out of grammatical order or 1') Partial match with some content loss, but wording in grammatical order 2) Contiguous phrase
S 1) Exact match, synonym, paraphrase, but wording out of grammatical order or 1') Partial match with some content loss, but wording in grammatical order 2') Non-contiguous phrase
Z Lost OR not recognizable

Table 5.3: Category codes for wh-items identified in MT output or lost in translation

To capture differences among MT engines' output and quality (independent of task results), we scored the individual wh-items by category code to measure translation accuracy and loss in the MT text proper. The category counts, accuracy rate, and loss rates are broken out in Table 5.4.¹³ There were no statistically significant differences between SMT (MT-2) and LBMT (MT-3) in terms of available wh-items to extract (total A, B, and Ss), but RBMT (MT-1) showed a significant loss in wh-items (Zs only).¹⁴

GT: كتبت ريم الميع: في قصر بيان ...
 RT: Reem Meeh wrote:
 at Bayan Palace...
 MT1: Reem wrote 'Lmceaa: [S]
 in a statement derelict,...
 MT:2 I wrote Rim almayai [الميع] : [A]
 in the short statement,...
 MT3: clerks move flowing :[Z]
 in castle demonstration/statement?...

Figure 5.6: Example of the same wh-item underlined in parallel sentences: the Arabic GT, English RT, and three MT outputs with category codes in square brackets

	A	B	S	Z	Prec.	Recall	Loss
MT1 Totals	67	51	20	18	.49	.43	.12
Who	21	17	12	6	.42	.38	.11
Where	34	15	2	5	.67	.61	.09
When	12	19	6	7	.32	.27	.16
MT2 Totals	91	49	9	7	.61	.58	.05
Who	29	19	7	1	.53	.52	.02
Where	41	12	1	2	.76	.73	.04
When	21	18	1	4	.53	.48	.09
MT3 Totals	67	75	4	10	.46	.43	.06
Who	21	26	2	7	.43	.38	.13
Where	33	22	0	1	.60	.59	.02
When	13	27	2	2	.31	.30	.05

Table 5.4: Results of category coding of wh-items in MT outputs

¹³ Let |A| stand for the # wh-items coded as A's. Precision was defined as $|A| / (|A| + |B| + |S|)$, Recall as $|A| / (|A| + |B| + |S| + |Z|)$, and Loss rate as $|Z| / (|A| + |B| + |S| + |Z|)$.

¹⁴ MT-2 showed higher precision and recall than the other two engines (As only). Details are presented by Vanni (2004).

We hypothesized that, if correct local word order were critical for wh-item extraction, then the LBMT (MT-3) would yield the weakest response rates. If word order however were not critical, then LBMT (MT-3) would outperform RBMT (MT-1) due to the latter's loss rate and match SMT (MT-2).

The task-based results by MT engine show subjects did statistically significantly better on two of the three metrics: they found more wh-items (higher recall) and they were misled by fewer non-wh-items (higher precision) in the statistical MT-2 output than in the output of the other two engines. The results did not yield a clear ranking of MT-1 and MT-3: the lexicon-based MT-3 did a better job in support of higher hit rates, while rule-based MT-1 did better with lower false alarm rates. Curiously, the miss rates across engines do not differ significantly: subjects fail to detect roughly the same proportion of wh-items in the output of each engine.

The task-based results by wh-type show subjects did statistically significantly better marking where- and who-items correctly than the when-items. They were least likely to miss the who-items. This contrasts with the MT results, where the miss rates were comparable across engines. Subjects were roughly equally likely to mark items incorrectly across wh-types. This contrasts with the MT results, where false alarm rates differed significantly.

To rank the MT systems on the basis of a single metric, we applied an approach from statistical decision theory: the three different performance rates – hit, miss, and false alarm rates – were combined with weighted cost estimates into a single loss function (Tate 2006). Based on several tests with hypothetical weights that we selected, the overall preference ranking of MT systems that persisted, where higher value (lower cost) is better than lower value (higher cost), was: MT2 > MT3 > MT1.

5.3.2.5. Correlating Task-based & Text-based Scores

The next phase of the evaluation was to ask how well text-based (DT) scores on task ii documents track the task-based (OT) results on these documents. Four automated text-based metrics were run on all machine-translated documents, to capture the MT engines' differences on full sentences. These results, presented in Table 5.5, are consistent with the text-based precision and recall on wh-items presented in Table 5.4; MT-2 ranks above the other two engines. However, the automated metrics do not provide a consistent picture for ranking MT-1 and MT-3: BLEU and GTM rank MT-1 well above MT-3, while METEOR and TER rank MT-3 slightly higher than MT-1.

System	BLEU	GTM	METEOR	TER
MT-1	.088	.529	.385	.221
MT-2	.187	.617	.524	.370
MT-3	.055	.453	.397	.233

Table 5.5: Automated metric scores on task ii document sentences by MT engine

Corr	Level	System	BLEU	GTM	METEOR	TER
P	system		.663	.468	.865	.86312
S	document		.211	.193	.242	.231
S	document	MT 1	.140	.147	.109	.235
S	document	MT 2	.134	.303	.298	.111
S	document	MT 3	.298	.323	.311	.182
S	document	Who	.253	.292	.320	.198
S	document	Where	.229	.185	.252	.251
S	document	When	.250	.158	.237	.331

Table 5.6: Text-based automated metrics correlated with task-based Hit Rates, for Pearson (P) and Spearman (S) correlations, at system and document level

The standard for meta-evaluation of new automated MT metrics has, until recently, been to aggregate autometric scores and human subjective judgment scores to the system level, and then to interpret the strength of their correlation as a validation of the former in terms of the latter.

We note that high correlations between automated metrics and task ii hit rates can be achieved when results are aggregated at the system level, as shown in the top row of Table 5.6. This is most evident for METEOR and TER metrics.

The correlations drop significantly when the text-based and task-based scores are paired at the individual document level. The modification, shown in the second row of Table 5.6 also yields correlations that are much closer to each other and have a much smaller range in values for the different automated metrics.

Pursuing this result further, when document-level scores are paired within MT systems, the table shows monotonic and significantly positive associations between hit rates and automated metrics, for all four metrics. Similarly the table shows significant relationships after grouping the data within Wh-type.¹⁵

These correlation analyses support the initial conjecture for task ii: subjects perform less well when given documents translated poorly by a weak MT engine, while they perform better when given documents translated by a stronger MT engine (Tate 2007).

5.3.2.6. Impact of Task-Based Approaches in MT Evaluation

Task-based approaches to MT evaluation seek to provide MT users with results that they can interpret relative to their own text-handling tasks. This section presented results of a large-scale DocEx experiment where subjects performed wh-item extraction on MT texts. The study found that the accuracy of their task responses yielded results consistent with a text-based measure of document translation-difficulty, derived from an automated MT evaluation metric. These results suggest that the task accuracy on wh-item extraction may, with further studies, be estimated from automated text-based metrics.

¹⁵ All table correlations are permutationally significant with p-values < .01 (Tate 2008).

5.3.3. Using United States Government Language Proficiency Standards for MT Evaluation

Authors: Doug Jones, Wade Shen, Martha Herzog, Sabine Atwell, Hussny Ibrahim, and Dan Ding

5.3.3.1. Introduction

The purpose of this section is to discuss a method of measuring the degree to which the essential meaning of the original text is communicated in the MT output. We view this test to be a measurement of the fundamental goal of MT; that is, to convey information accurately from one language to another.

We conducted a series of experiments in which educated native readers of English responded to test questions about translated versions of texts originally written in Arabic and Chinese. We compared the results for those subjects using machine translations of the texts with those using professional reference translations. These comparisons serve as a baseline for determining the level of foreign language reading comprehension that can be achieved by a native English reader relying on machine translation technology. This also allows us to explore the relationship between the current, broadly accepted automatic measures of performance for machine translation and a test derived from the Defense Language Proficiency Test, which is used throughout the Defense Department for measuring foreign language proficiency. Our goal is to put MT system performance evaluation into terms that are meaningful to US government consumers of MT output.

5.3.3.2. Defense Language Proficiency Test

The official Defense Language Proficiency Test (DLPT) is constructed according to rigorous, well-established testing principles to measure the foreign language proficiency of military personnel trained by the Department of Defense and several other United States government entities. Developed and improved over the course of several decades, the DLPT is focused on the language proficiency standards recognized and used throughout the government.

In support of studies that preceded GALE to determine the accuracy vs. utility of MT, we constructed a variant of the DLPT, adhering to the test construct and design principles, but making modifications to measure the quality of machine translations. This test became known as the DLPT – Standardized Translation Assessment with Reference, or DLPT-STAR.

Earlier DARPA work in MT evaluation incorporated (1) an informativeness measure, based on answers to comprehension questions; (2) fluency, a measure of output readability without reference to a Gold Standard; and (3) adequacy, an accuracy measure that did refer to a Gold Standard translation (White OConnell 1994). Later MT evaluation found fluency and adequacy to correlate well enough with automatic measures (BLEU). Since comprehension tests are relatively more expensive to develop, the informativeness test was not used in later MT evaluations, such as those performed by NIST from 2001-2006. In other work (Voss and Tate 2006), and as described in the previous section, task-based evaluation has been used for MT evaluation. These

techniques measure human performance by exhaustively extracting ‘who’, ‘when’, and ‘where’ type elements in MT output.

The DLPT-STAR also uses such factual questions, but this is not the only question type. The DLPT-STAR test items focus on the elements in the original text that are most characteristic of its proficiency level as defined by the Interagency Language Roundtable (ILR) Language Proficiency Scale. For example, Level 3 texts may have test questions about abstract linguistic formulations, hypotheses, or supported opinion essential to the meaning of the text. The test construct of the DLPT-STAR is based on the ILR scale, which is used throughout the United States government, so it provides Defense Department decision makers with results they can readily interpret. A description of a text classification scheme based on the ILR scale can be found in Child (1993). Some key points about the ILR scale for reading at Levels 1 through 3 are shown below:

Level 1 texts: contain short, discrete, simple sentences; generally pertain to the immediate time frame; often written in an orientational mode; require elementary level reading skill. For example, newspaper announcements are typically written at Level 1.

Level 2 texts: convey facts with the purpose of exchanging information; do not editorialize on the facts; often written in an instructive mode; require limited working proficiency. For example, newswire articles are often Level 2.

Level 3 texts: have denser syntax and highly analytic expressions; place greater conceptual demands on the reader; often written in an evaluative mode; may require the reader to ‘read between the lines’; require general professional proficiency. For example, newspaper opinion and editorial articles may be written at Level 3.

5.3.3.3. Experimental Design

The DLPT-STAR measures the quality of machine translation output in terms of readers’ comprehension of the text as reflected by the accuracy of their responses to short-answer constructed-response questions. Subjects participating in the test were allowed five to eight hours to complete the experiments, depending on the specific test. Approximately 50 participants were recruited from MIT and the surrounding community for each test. All were native readers of English. The experimental tests were delivered by computer. Subjects were shown each text and its associated questions simultaneously. They were allowed to work at their own pace. The interface of the delivery software allowed them to navigate between texts and take breaks as needed.

Each subject’s responses were rated against a predetermined scoring protocol. The same scoring process, protocol, and standards were used for all responses, regardless of whether the subject used MT output or the reference translation (the control case). The raters were blind to the conditions of the responses they scored.

We expected that, if the reference translation were better than the MT output, the reference should yield a higher percentage of accurate responses than the MT.

5.3.3.4. Initial Study – 2004

The first DLPT-STAR test was based on authentic Arabic texts at ILR Levels 1, 2, and 3, with short-answer questions at the level of the texts. Texts were identified by a team with substantial experience in ILR text rating and DLPT development. The documents were presented in two conditions of English translation: (1) professionally translated into an appropriate English equivalent, and (2) machine translation output from state-of-the-art MT systems; this was often quite garbled.

This experiment showed that, when reading MT output, native readers of English could generally attain or surpass the passing score on Level 1 texts with 75% comprehension and on Level 2 texts with 76% comprehension.¹⁶ However, with Level 3 texts, the result was 51% comprehension, well below the passing score. Comprehension of the Gold Standard reference translation was 95% at Level 1, 91% at Level 2, and 79% at Level 3, as shown in Figure 5.7. The results are presented in greater detail elsewhere (Jones *et al.* 2005).

The lower results for Level 3 on the reference translation reflected the higher degree of difficulty of the original texts. Subjects may not have frequently read this kind of prose, even in their native language. However, we could not determine whether the somewhat weak performance at Level 1 on the MT output was due to systematic deficits in MT performance at Level 1 or to a mismatch between the texts and MT capabilities. Subsequent experiments in the GALE program were designed to address the ILR scale levels while using materials better suited for MT processing.

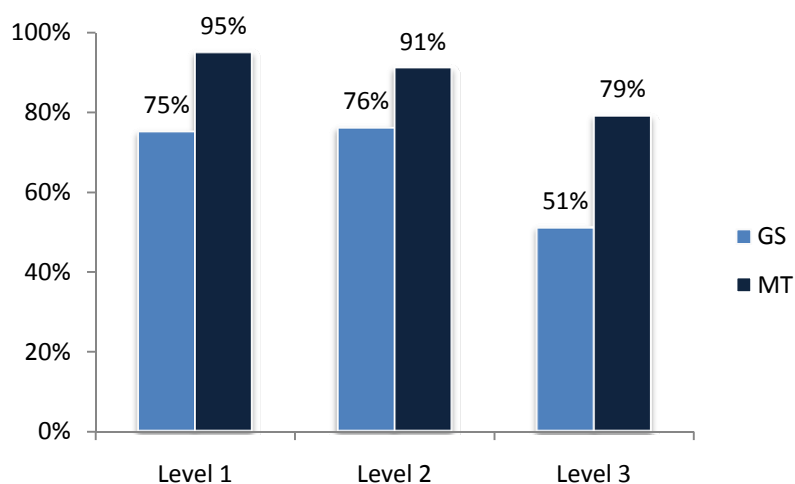


Figure 5.7: First DLPT-STAR Results

5.3.3.5. Second Test – Arabic MT

In 2006 during Phase 1 of GALE, we created a new variant of the DLPT-STAR, using materials specifically created to test the capabilities of the MT systems. We used

¹⁶ In line with many language proficiency testing practices, we set 70% as a passing grade.

the DARPA GALE 2006 evaluation data sets, used by several research sites for testing MT algorithms. We arbitrarily merged the MT output from three GALE performers.

The ILR levels of the documents ranged from Level 2 to Level 3; DARPA GALE 2006 data did not contain any texts that could be rated as Level 1. The absence of Level 1 documents was not accidental; the kinds of texts associated with Level 1, such as newspaper announcements, advertisements, weather reports, etc., have not been part of the typical training or test data for MT evaluation and would not be of interest to the Department of Defense.

To partially compensate for the lack of authentic Level 1 material, we constructed questions about Level 1 elements (including personal and place names) found in Level 2 and 3 texts. A standard DLPT would use texts rated as Level 1, resulting in more variety in topic and testing points. It would also ask Level 1 questions only about Level 1 texts, Level 2 questions about Level 2 texts, etc. Our method of compensation is only partial, because it does not show the true performance of MT systems on Level 1 documents. It is possible that the performance we have seen at Level 1 may be atypical. Training of the systems on Level 1 texts would be needed to provide answers.

We selected approximately half of the DARPA GALE 2006 evaluation material for our test. There were twenty-four test documents, with balanced coverage across four genres: newswire, web data, broadcast news, and broadcast conversation. Our target was to have at least 2500 words for each genre; we exceeded this slightly with approximately 12,200 words in total for the test. We constructed an exhaustive set of questions for each document, approximately 200 questions in all. See Jones *et al.* (2007) for additional details.

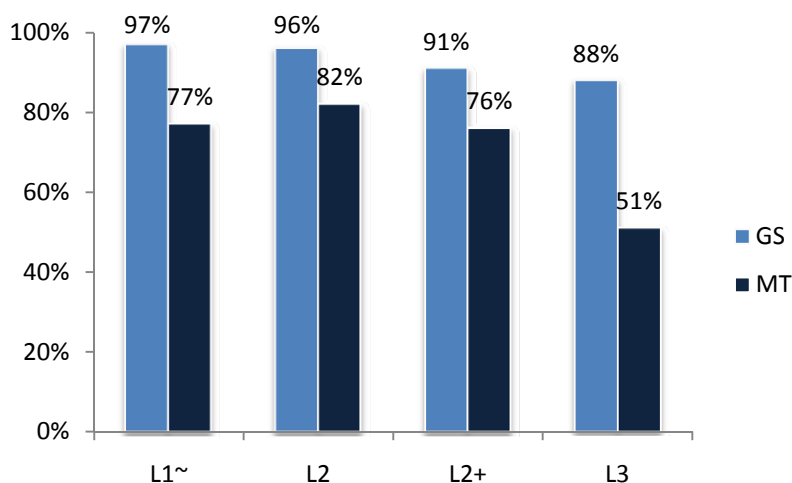


Figure 5.8: DLPT-STAR Results by ILR Level

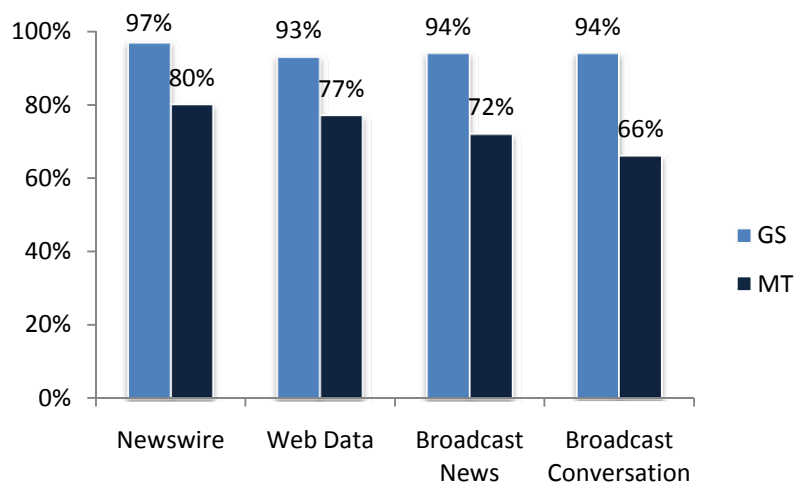


Figure 5.9: DLPT-STAR Results by Genre

The overall comprehension results for MT and for the Gold Standard (GS), professionally-produced reference translations, are shown by ILR level in Figure 5.8. The results by genre are shown in Figure 5.9.

The pattern of results for the second test using the DLPT-STAR was similar to that observed in the initial study:

- Level 1 performance was both lower than Level 2 and lower than expected.
- Level 3 was lower than Level 2, as expected.

Comprehension results for MT output were 77% for Level 1, 82% for Level 2, 76% for Level 2+, and 51% for Level 3. Gold Standard (GS) results ranged from 97% for Level 1 to 88% for Level 3.

Further analysis showed the low performance of MT at Level 1 to partially stem from the systems' incorrect translations of personal names, although these were essential to correctly answering many Level 1 questions. We observed that the name questions had 71% comprehension accuracy, compared with 83% comprehension for Level 1 items testing language features other than personal names. We should reiterate that MT systems have typically not had Level 1 documents as training data. Unfortunately, as long as this imbalance remains, the DLPT-STAR cannot be expected to show a consistent trend in performance across the ILR levels. Whether the addition of appropriate training data would affect performance is an open question.

Earlier (Jones *et al.* 2007), we also examined the relationship between comprehension rates and translation error. We compared comprehension rates with HTER, which counts the edits required for MT output to contain all, and only, the information present in a Gold Standard reference. A linear regression showed that subjects lose about

12% in comprehension for every 10% of translation error, with an R^2 value of 33%. The low correlation suggests that the comprehension results are measuring a somewhat independent aspect of MT quality. We feel this distinction is important. HTER does not directly address the fact that not all MT errors are equally important and that texts contain inherent redundancy that readers use to answer questions.

5.3.3.6. Using the DLPT-STAR for multiple MT systems in NIST's MT 08

Although we believed the overall design of previous DLPT-STAR tests was sound, we built the next test taking care to select documents across a broader range of topics and data sources. Also, previous tests compared machine translation output with professional reference translation output. The present test was designed to contrast different MT systems, using GALE phase 2 materials and administering the DLPT-STAR in the NIST 2008 OpenMT evaluation.

We selected ten Arabic-to-English MT systems and ten Chinese-to-English MT systems that were part of the NIST 2008 OpenMT evaluation run on the progress test set. We used all ten Chinese Systems that participated in the NIST evaluation. We chose ten Arabic systems, from the top, middle, and bottom of the range of BLEU scores attained in the NIST evaluation. In order to have a manageable number of conditions per test, we ran five MT systems and one GS reference translation with 100 subjects at a time.

We tested two of the GALE genres, structured text (Newswire) and unstructured text (Webdata). Overall, Arabic performance was higher than Chinese, and Newswire performance was higher than Webdata. The BLEU scores and the DLPT-STAR comprehension results are shown in Figure 5.10 and Figure 5.11 for two of these cases.

Figure 5.10 shows Arabic Newswire results, the highest comprehension results. The systems are sorted according to DLPT-STAR comprehension results and are shown by the gray bars. The white bar at the right indicates performance on the reference translations. The black bars indicate the BLEU score for the progress test set for each of the systems. System 'MT-A7' has the lowest comprehension rate and the lowest BLEU score. However, the highest BLEU score belongs to 'MT-A3', which is in the middle range of comprehension results. The highest scoring DLPT-STAR score is associated with a relatively low BLEU score. For comparison, Figure 5.11 shows the results for Chinese Newswire.

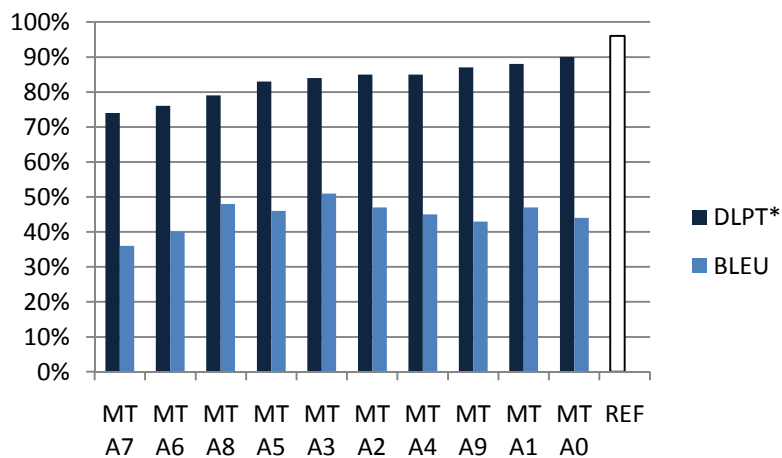


Figure 5.10: Arabic Newswire

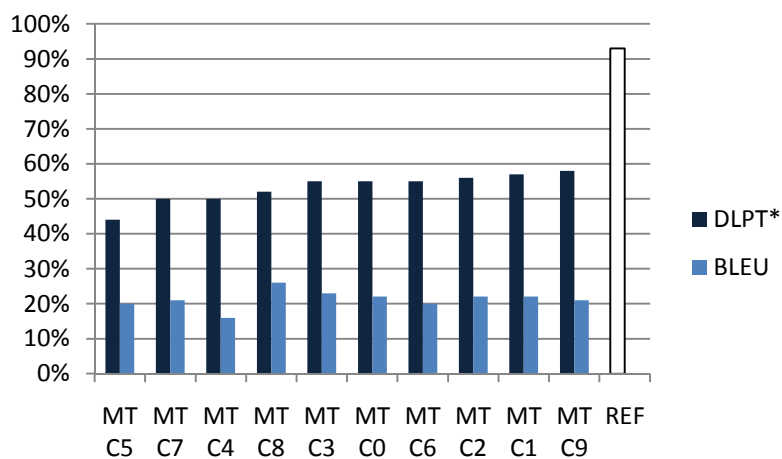


Figure 5.11: Chinese Web data

System-level correlation is shown in Figure 5.12. Each point indicates the overall BLEU score and overall DLPT-STAR score for one MT system in one condition. Since there are ten systems, two languages, and two genres, there are forty points in the cloud in Figure 5.12. At this broad level, the R^2 value is 74%.

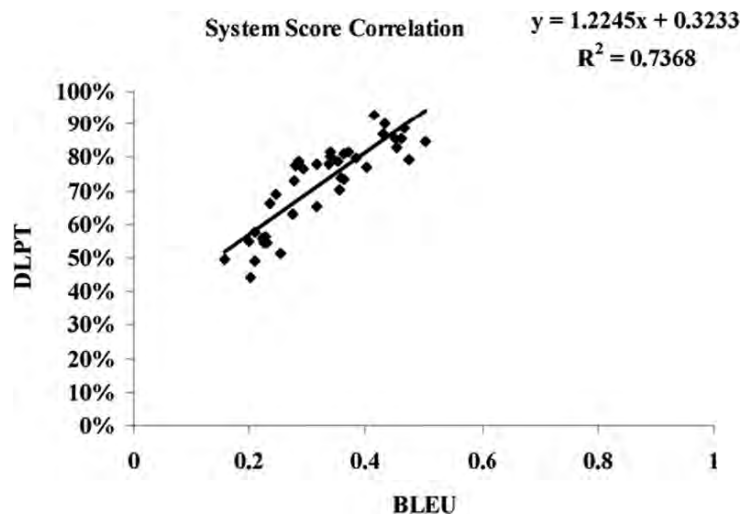


Figure 5.12: System level correlation

5.3.3.7. Impact of Language Proficiency Standards in MT Evaluation

Overall, we have found that Level 2 texts are a natural fit for successful machine translation. The MT systems seem to show an ability to convey concrete factual information that can be retrieved and used by the reader who does not need to understand the style, tone, or organizational pattern used by the writer.

Over the course of this series of experiments, we have observed increasing levels of comprehension demonstrated by readers of MT output. However, each experiment was run with a new DLPT-STAR test, making comparisons difficult. Now that we are able to sequester the DLPT-STAR as part of the NIST OpenMT progress test set, we will have an opportunity to observe trends with the same test material.

5.3.4. Concluding Remarks on Task-based Evaluation

Authors: Clare Voss, Doug Jones, and Martha Herzog

Both research studies discussed in this section report on human subjects performing language tasks using texts that were derived from the original foreign language source. The degree of distance from the original text depended on the quality of the translation, whether professional reference translation or machine translation output. The language tasks for the subjects were related to the communicative intent of the original text. That is, communication of the text's message was evaluated by requiring subjects to use all or portions of the message for a purpose. These two sets of studies approached the evaluation of communicative intent from different directions. Specifically, the first study focused on identifying the core factual information comprising the text to determine the relative accuracy of each system's output. The second study focused on determining

whether subjects using machine translation output have adequate material to answer content questions about the most significant information in the original text. The first study examines the building blocks of a text, while the second examines the purpose for writing it. The interaction between subjects and text demonstrates the degree to which the original communicative intent is conveyed in the output. The subjects' accuracy in performing the tasks demonstrates the usefulness of the output. Most importantly, both studies move the research closer to evaluating real-world use of machine-translation systems.

5.4 GALE Machine Translation Metrology: Definition, Implementation, and Calculation

5.4.1. GALE Machine Translation Metrology: Introduction

Authors: Mark Przybocki, Audrey Le, Gregory Sanders, Sebastián Bronsart, Stephanie Strassel, and Meghan Glenn

One way to evaluate the quality of a machine translation is to note what edits (changes) a human editor will make to correct the meaning of the translation and to make the translation understandable (possibly also to make the translation reasonably fluent). We refer to that editor as a *post editor*. Ideally, the post editor will be bilingual, with a high-level understanding of both the source language and the target language (the language translated into), with the target language as his/her strongest language. However, post editing can be performed by a monolingual editor who compares the machine translation (MT) output to one or more carefully produced reference translation.

The GALE program uses the metric of *edit distance* to evaluate machine translation performance. As in many manual assessments of human language technologies, post editing can be implemented in a variety of ways. For GALE, the implementation of edit distance requires monolingual human post editors that refer to a carefully produced reference translation, to make the minimum set of edits that completely corrects the meaning and makes the translation as understandable as the reference translation.

The TERCOM scoring software (Snover 2006) automatically compares the original (unedited) MT output to the post-edited version, and counts the number of changes. We refer to the number of such edits, divided by the number of words in the reference translation as the “Human-mediated Translation Error Rate”, or HTER (Snover 2006).

This section will focus on HTER and post editing, where we will define HTER and describe its use in the GALE evaluation of MT. The entire post-editing process will be detailed, including the recruitment and training of editors, the software tool and guidelines used for facilitating the edits, the workflow management system and quality control (QC) methods employed, and how the output of the post-editing process is used for the final calculation of HTER. Where appropriate, this section will cover the interaction between the post-editing process and HTER scoring. GALE post editing is the

largest manual scoring effort of MT metrology being employed in any of the current NIST coordinated MT technology evaluations.

This section also addresses the challenges of editor consistency, and how GALE evaluations use control documents to allow for year-to-year editor consistency measurement. It describes the year-to-year changes made to the post-editing process, including improvements to the editing guidelines, added flexibility to the post-editing software, and streamlining the post-editing process. We also include a section that describes how MT output is selected and assigned to particular editors, such that all GALE system translations are handled equivalently. Finally, we describe the overlap of work between particular editors and how this information can be used to identify consistent (consistently good or bad) editors.

Although human post editing of present day MT outputs is expensive, we have probably come some distance from the situation in the nineteen-sixties, when the [controversial] ALPAC report observed that,

. . . when after 8 years of work, the Georgetown University MT project tried to produce useful output in 1962, they had to resort to post editing. The post-edited translation took slightly longer to do and was more expensive than conventional human translation. (ALPAC 1966)

5.4.2. GALE Machine Translation Metrology: From TER to HTER

Authors: Mark Przybocki, Audrey Le, Gregory Sanders, Sebastián Bronsart, Stephanie Strassel, and Meghan Glenn

We begin by explaining the adaptation of TER to a human-mediated evaluation measure, HTER. As described in Section 5.2.2.4, TER is defined as the measure of edit distance when editing the original MT output to exactly match a human reference translation. The scoring software simply compares the original MT output to the reference translation. But it is not necessary for MT system output to match a reference translation word-for-word in order to convey the complete meaning of the translation. Synonyms, phrasal reordering, and substituting pronouns or simplified forms of noun phrases for already defined proper nouns are just a few examples of acceptable alternatives.

For example, suppose the reference translation reads, “*His black car had a dented bumper, and when the car was driven down the street, the observers called us to report that it was on the move.*” In this case, there is no problem if the MT output reads, “*His black car had a dented bumper, and when the black car was driven down the street, the observers called us to report that the black car was on the move.*”

Correspondingly, the GALE metric for MT evaluation is not a measure of the number of edits required to match the reference translation text TER, nor is it a measure of the number of word matches between the MT output and the sets of reference translations (as in BLEU or WER). Rather, it is a measure of the minimum number of edits required to

match the meaning of the reference translation and make it equally understandable (HTER).

Deciding which textual alternatives alter the meaning or impair the understandability of the translation requires human judges. For the GALE evaluation of MT system performance, human editors compare the original MT output to the reference translation and manually modify the MT output using as few edits as they can manage to make the MT output contain the same meaning as the reference translation and be equally understandable. We refer to this process as *post editing*.

HTER (Human-mediated TER) is a measure of edit distance between the original MT output and the post-edited version of the original MT output. HTER is the official metric used to evaluate GALE MT system performance. Thus, where TER measures the edit distance to change the machine translation output into a human reference translation, in contrast HTER uses the same scoring software but measures the edit distance to change the same machine translation into the final post-edited version.

In evaluation, output from multiple MT systems processing the same input, are compared. Thus, it is necessary in the calculation of HTER to divide the number of edits for each system by the number of words in the reference translation rather than in the original system translation, so that the denominator is the same across each system. Using the token count from the reference as our denominator serves as a normalizing factor, allowing for direct comparison of HTER scores. Without normalization, systems could reduce their error scores by simply outputting superfluous text. For example, if a system translation was “They *traveled to Mexico*,” an example of superfluous text might be “*The group undertook a trip and traveled to the country of Mexico.*” Now consider in each case if the reference referred to a trip to Spain. In each example there would be one edit (substituting Spain in place of Mexico). The first example would have an HTER score of 25% error (1 error divided by 4 token words), while the second example would have an HTER score of 8% error (1 error divided by 12 token words).

The following sections explain the process of HTER calculation for the GALE program. The required infrastructure is reviewed, including the editing guidelines, the recruitment and training of editors, the editing software, and the workflow system required to manage the post-editing effort. We describe the methods employed to accommodate for editor inconsistency and fatigue which is a part of every human effort that imposes a heavy cognitive load.

5.4.3. GALE Machine Translation Metrology: Post-editing

Authors: Mark Przybocki, Audrey Le, Gregory Sanders, Sebastián Bronsart, Stephanie Strassel, and Meghan Glenn

5.4.3.1. Post-editing Guidelines

The definition of HTER centers on what edits are to be made. The editors compare the MT output to a correct reference translation, and edit the MT output correspondingly. To date, the translations to which GALE has applied the HTER metric have all been translations into English (the target language) from two source languages, Arabic and Chinese. Thus, English was always the language being edited.

Four goals were discussed at length during the design of the editing guidelines. First and foremost is that the edits should completely correct the meaning. The second goal is that the editors should find the minimum set of edits that will completely correct the meaning. These first two goals amount to the underlying idea of the HTER metric. The third goal is that the edited translation needs to be understandable English, assuming the source material is understandable. This third goal is an important constraint on the editing process. A fourth goal that was discussed, and ultimately rejected, was to ask the editors to produce fluent English, if the source-language material was fluent.

5.4.3.1.1. The Process of Creating the Guidelines

The Post-editing Guidelines (NIST 2007) reflect the goals that were just mentioned. All parties to the GALE program agreed with the goal of completely correcting the meaning, but when various parties in the GALE community post edited several example translations, there were differences in what those various editors had corrected. As the parties discussed those differences, some decisions emerged as to the desired edits. NIST and the LDC created the first full versions of the post-editing guidelines in order to capture those decisions; the guidelines increase the level of agreement among editors regarding what constitutes a difference of meaning, and thus increase the level of agreement about the edits required to completely correct the meaning. The guidelines were refined through an iterative process of using successive versions of the guidelines to post edit example translations, looking at the edits that resulted, and modifying the guidelines appropriately. The guidelines continue to be reviewed after each phase of GALE evaluation, allowing post editor comments to be considered.

Through that iterative process, it became apparent that some editors were distinctly cleverer than others at finding ways to correct the text with fewer edits. Ways to correct the text with fewer edits were addressed at several places in the guidelines.

5.4.3.1.2. Displaying the HTER Value to Help Minimize the Number of Edits

Minimizing the number of edits was also addressed in the recruitment and training of the editors; seeking editors who would find it interesting to find a minimal set of edits. The main tool assisting editors to understand what constitutes fewer edits has been the post editing software itself, which was modified to display (see upper right window pan in Figure 5.13) the HTER value for the segment currently being edited, an HTER value that resulted from the edits just made. This modification to the tool was effective because it made it easy for editors to try various alternative edits, with immediate feedback about their effect on the HTER value. In practice, having the editors consult the displayed HTER value for the segment has proven effective.

5.4.3.1.3. The Post Editor's Role

The job of the MT post editors may be summarized as follows: “Make the MT output have the correct meaning (the same meaning as the reference translation), using understandable English, in as few edits as possible.”

The guidelines ask the editor to keep in mind that if the MT output adds/inserts information that is not present in the reference translation, those additions should be removed. Likewise, if the MT output omits information that is present in the reference translation, the missing information should be added.

Although the editors work on one segment at a time, the tool displays all the preceding (already edited) and following (not yet edited) segments in the current document. This allows the editor to see the complete context of the document, both as the reference translation and as the system output being edited. The guidelines ask the editor to keep in mind that the goal is to correct the meaning of the document as a whole, and that the MT systems are free to rearrange words and phrases over long distances (to/from other segments in the document). The editor is not to gratuitously alter those rearrangements if they do not affect the meaning. Without the display of context, judges could have difficulty identifying acceptable rearrangements. In practice, GALE data identifies the sentence segmentation for the systems, practically eliminating rearrangements.

Finally, we ask the editor to avoid wholesale copying of the reference.

5.4.3.1.4. A Condensed Version of the Post-editing Guidelines

- Make the MT output have the same meaning as the reference human translation: no more and no less.
- Make the MT output be as understandable as the reference. Similarly, try to make the MT output not be more or less ambiguous than the reference.
- Punctuation must be understandable, and sentence-like units must have sentence-ending punctuation and proper capitalization. Do not insert, delete, or change punctuation merely to follow traditional rules about what is “proper.”
- Capture the meaning in as few edits as possible using understandable English. If words/phrases/punctuation in the MT output are completely acceptable, use them (unmodified) rather than substituting something new and different.

In case of conflicts among the four rules above, consider them to be ordered by importance. For example, never sacrifice the goal of correcting the meaning just to minimize the number of edits.

5.4.3.2. Post-editing Software

The Post-editing Software is a Java application that NIST designed and programmed. A strong collaboration between NIST and LDC allowed for frequent improvements of the software over time. Version 1.2.2 of the post-editing software was used for GALE evaluations at the time of this writing.

In order to ease the process of distributing the software to the annotators, a single JAR file bundles all of the required libraries. This includes the TERCOM (Snover 2006) library that computes the HTER score and provides a visual representation of the modifications made to the original MT output.

5.4.3.2.1. The MTF File Format

The Machine Translation File format (MTF) is the primary exchange format used throughout all GALE post-editing evaluations. MTF is based on XML: a DTD file defines the set of elements and attributes that may appear in a file, their respective relationships, the possible attribute values, and so forth. NIST chose to base MTF on XML for the following reasons:

- Given a DTD, any XML file can be validated using one of the many XML validating tools available.
- XML handles Unicode encodings.
- XML syntax is fairly simple and allows for easy manual editing of the contents.

The root element of an MTF file is a container for one or more documents, which in turn contain one or more segments. Usually, one segment corresponds to a single sentence. Several attributes are used to identify the data contained in the file, allowing tools to automatically parse the contents of a set of MTF files.

Because members of each editing team work successively on the same file (see Sections 5.4.3.3.2 and 5.4.3.3.4), the MTF format embeds facilities for the team members to communicate. One such feature, the *Request For Review* attribute, is used when an annotator decides to notify a supervisor of any inconsistency in the guidelines or, more realistically, in the data being edited. For example, an annotator may want to report that a reference sentence requires a review if he/she thinks there might be an error in the reference translation.

5.4.3.2.2. The Post-Editing Interface

The main window (see Figure 5.13) is divided into three main panels. The leftmost panel contains the reference translation for a given document. The middle panel is where the translation of the document is edited. The rightmost panel is used to display the HTER representation (as described in Section 5.4.3.1.2) and the original version of the current segment.

This leftmost panel (with the reference translation) is divided into three text boxes: the top box displays all previous segments for this document, the middle box displays the current segment, and the bottom box displays all succeeding segments for the document. Some reference words, or groups of words, may have associated alternate translations. If so, these words are highlighted and alternate translations are shown via a drop-down list that is invoked by hovering the mouse pointer over the highlighted text.

The middle panel (with the edited translation of the document) is also divided into three text boxes, containing respectively all the previous segments, the segment currently

being edited, and all the succeeding segments yet to be edited. Additional buttons are provided to allow the annotator to flag the current segment, in case it cannot be edited directly.

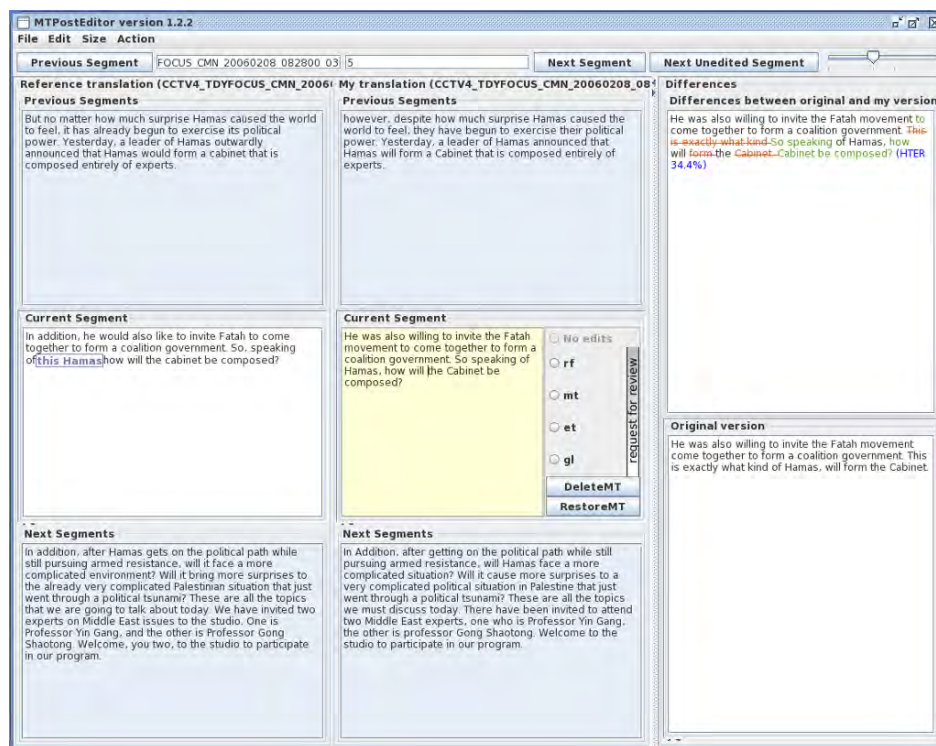


Figure 5.13: Main window of the post-editing tool

5.4.3.3. The Evaluation

5.4.3.3.1. Recruitment and Training of Post Editors

Linguistic Data Consortium recruits Philadelphia-area native English speakers with training in copy-editing, proofreading, creative writing, or journalism. First-time applicants residing outside of the region are excluded because the training process requires a visit to LDC in Philadelphia. All candidates receive by email a pre-test “kit,” containing a brief set of instructions, the annotation tool and the test file to edit. The test file consists of eleven sentences which LDC selected from GALE Phase 1 machine translations that represent the two source languages and multiple genres and machine translation systems. The eleven sentences were also selected to represent a range of editing difficulty, so that the first few sentences of MT output require fewer changes, and the last sentences require more – and possibly more creative – edits.

Managers score and review the test kits carefully, looking for a basic post-editing aptitude in the test kit responses: primarily, correct spelling, incorporation of the full meaning of the Gold Standard reference in the test edits, and no extraneous information.

After examining the pre-test results and eliminating outlier candidates, LDC invites qualified applicants to LDC's offices for an intensive group training session.

The training session focuses on the program goals and the MT post-editing guidelines developed jointly by NIST and LDC. Managers display a set of editing examples to demonstrate editing possibilities and pitfalls. Following the training session, applicants re-edit the test kits so that managers can observe what they would do differently after learning more about the task. Those who continue to produce edits that convey the same meaning as the Gold Standard translation and who make only necessary changes to the MT are selected for the project.

Many editors return for multiple cycles of post-editing within the GALE program. Returning editors are required to attend a re-training session at LDC. The re-training session focuses primarily on the same points as the original training session, but managers provide more time for questions and examples. In addition, the group works through multiple examples together. Editors are also encouraged to come to LDC for "office hours," or regular opportunities during the course of a project to sit down with a manager to discuss editing questions in general or review specific kits in detail.

Before beginning work on GALE evaluation data, editors read the guidelines carefully and complete a starter kit, a set of documents that reflect attributes of the data for the current phase. The starter kit reinforces their knowledge of the post-editing rules and allows editors and LDC staff to solve technical problems. It also offers managers another opportunity to evaluate editors and to answer many task-related and procedural questions before the project begins in earnest.

5.4.3.3.2. LDC Workflow and Quality Control

Editors work remotely, accessing post-editing assignments through a web-based workflow management site, which LDC developed for this task. The workflow system assigns kits, or files containing approximately 1200 words of translated material, to editors. Editors are assigned the role of first pass or second pass editor at the beginning of the project. The primary role of a first pass editor is to perform the initial editing of the MT output, finding the minimal number of edits required to capture the complete meaning of the reference translation. The primary role of the second pass editor is to review and check the work of the first pass editor. The second pass editor determines if the meaning of the edited MT output matches the reference. If correct, the editing is left alone. The secondary role of the second pass editor is to determine if appropriate modifications can be made in order to reduce the HTER score. Second pass editors are paired with first pass editors, to form a team. In most cases, two editors will be paired for the duration of a project. Translations are reviewed by two independent editor teams of first- and second-pass editors.

The workflow system serves both the editors and the LDC managers. LDC managers are able to view and manage users and assignments, as well as, backup the project through the system. Assignments, the kits, for each editor team are loaded into the workflow system at the start of the project. Editors check out kits from this system, working with one kit at a time. After a first pass editor checks in a completed kit and no problems are identified by an automated scoring process, the kit is automatically assigned to the second pass editor.

The workflow system was designed as a central information resource for editors. Here they can view their file assignment lists, verify expected payment per kit, and find links to other project resources. Editors are also able to check the status of kits in their queue, and monitor their first- or second-pass partner's progress. Second-pass editors benefit from a "nudge" option in the workflow system, which allows them to send a polite reminder to their editing partner that they are waiting for data.

LDC quality control measures are in place to catch careless errors or alert managers to potential problems during the editing process. In addition to managing file assignments, the workflow system supports such quality control measures. When an editor checks in a file, the workflow system triggers various automatic and manual LDC quality control mechanisms. For example, the first kit submission of every editor is flagged and held in a separate queue until approved by a manager. Scripts automatically score and check incoming kits to identify potentially problematic kits; these include kits with high HTER scores or with unedited segments. LDC managers also spot-check kits daily, and provide feedback to editors accordingly, in order to satisfy the quality requirements for the project.

5.4.3.3.3. Reducing the Cognitive Burden

As with most (possibly all) manual evaluations of translation quality, the difficulty of the task presented to the human judges has a direct bearing on the quality or consistency of their effort. This section describes specific efforts that were made to reduce the cognitive burden placed on the editors.

As was mentioned in the previous Section, data units or "kits" that are to be edited by the editors are kept to a reasonable size. For the first GALE evaluation of MT performance each kit contained approximately 1500 words of translated text to be edited. Post-evaluation editor comments led to reducing the kit size to approximately 1200 words. While post editing is performed remotely (left to the editor's choice) and the editors are under no scrutiny to work through a kit absent lengthy breaks, we have estimated that an editor spends approximately two to four hours editing each kit. The goal was to be sure kits were small enough to finish one kit per day. Kit size was a primary consideration in defining which documents were grouped into a specific kit.

A second consideration in the kit definition process was the original source language. We have observed through automatic and manual measurements of MT quality that the English translations of Arabic source data tend to be better than those from Chinese source data. With that in mind, kits were created as to mix, and preferably alternate between, documents that originated in the two languages. The thought being that the lower MT quality would require more thought, more careful editing, to achieve the post editing goal. This second consideration was not always possible. For example in the first MT evaluation during phase 3, only one team participated and only the Arabic source data was processed.

The third consideration designed to reduce the cognitive burden on editors related to mixing the genres in each kit. To the extent possible NIST sought to mix audio and text sources and within that grouping NIST sought to interleave translations from each of the four genres (text from newswire and web; audio from broadcast news and broadcast conversations). Again, through automatic and manual evaluations of MT performance by

genre, it has been observed that system performance was generally better for the structured sources when compared to the unstructured sources (translations of newswire data were generally better than translations of web data, for both languages; and translations of broadcast news data were generally better than translations of broadcast conversations, for both languages).

And finally, as was mentioned in Section 5.4.3.1.4, a one-page reminder of the post editing goals was produced as a quick reference guide. The full post editing guidelines are too comprehensive to expect editor memorization. This one-page reference guide emphasized the order of priorities and general philosophy behind the editing process and served new editors as a tool that could quickly be committed to memory.

5.4.3.3.4. Post-editing Protocols – Teams: first pass editors and second pass reviewers

This section describes the post-editing protocol used in the GALE program and the changes that were made in later phases of GALE such as to combine the first pass editors and second pass reviewers into a team, streamlining the post-editing process.

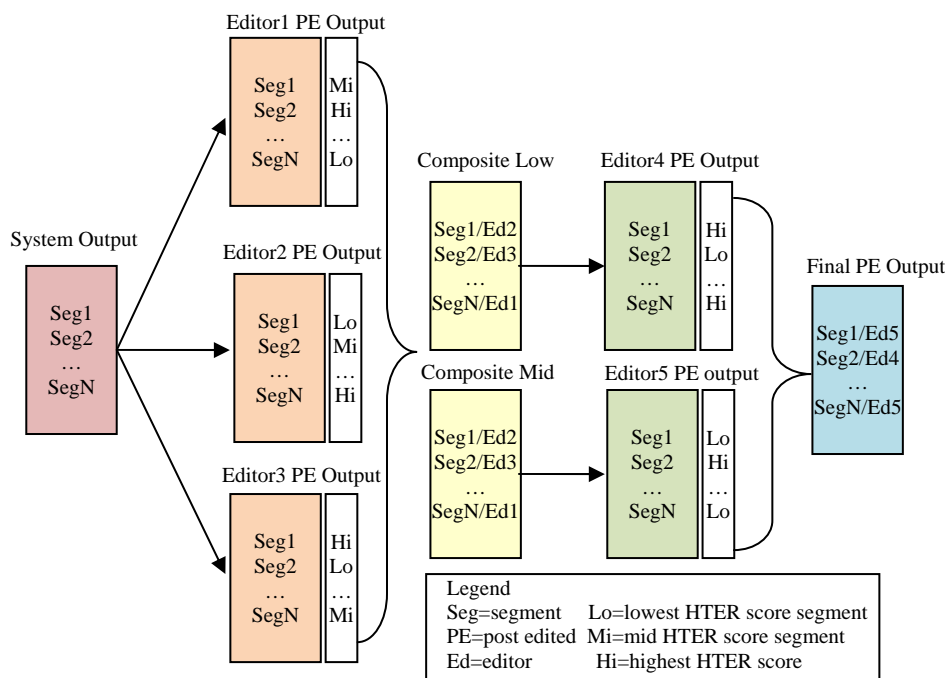


Figure 5.14: Diagram outlining the post-editing protocol used in GALE phase 1

We designed the GALE post-editing protocol to maximize repeatability and improve validity by employing a multi-stage process.

Figure 5.14 summarizes the protocol used in the first year of GALE. In the first year of GALE, the editing process involved three truly separate stages: first-pass editing, second-pass reviewing, and a final stage in which a composite document was assembled and scored. In the first stage, three independent first-pass editors post edited each

system's output translation. The result was three first-pass post-edited versions of the same document. The HTER score was computed for each segment of each of those three versions.

In the second stage, the two composite outputs were reviewed by different second pass editors who acted as a reviewer and made additional (or different) edits and corrections if needed. The focus of the reviewer was ensuring that the meaning of the post-edited translation matched the meaning of the reference translation. Usually the editors who served as reviewers were more experienced editors. The second pass editors were primarily concerned with ensuring the accuracy of the edits. This stage resulted in two second-pass post-edited versions. The HTER score was computed for each segment of each of the two second-pass versions.

In the third stage, a final composite output was created from the two second-pass post-edited versions of each segment by choosing the version with the lower HTER score for each segment. The final HTER value is computed over that final composite output.

We turn now to elaborating on the rationale for this process. For HTER to be meaningful, the post-edited output must be consistent and accurate. Accurate is defined as (1) completely correcting the meaning, (2) being as understandable as the reference, and (3) using the minimum number of edits necessary to accomplish those two goals. Consistent is defined as repeatable, which in practice means different editors would make similar edits on any given system output. However, the post-editing task is highly subjective despite the fact that editors have gone through a rigorous training process. Thus, multiple editors were used to post edit each system output, in an attempt to ensure editing consistency.

In the first stage, for practicality reasons (time and cost) we chose the number of first-pass editors to be three although a higher number would have been more desirable. As has been mentioned, the post-edited translation should convey the same meaning as the reference translation. This does not mean the post-edited translation should have exactly the same wording as the reference translation. Using three independent first-pass editors generated a wider range of alternative ways to edit the translation so as to minimize HTER. The second-pass review editors ensured that the meaning was not compromised. The reviewer is necessary because minimizing HTER conflicts with improving the meaning of the translation.

Making as few edits as possible during the editing process is important to a meaningful HTER score (recall that the idea of HTER is that it is a measure of the minimum edits required to correct the MT output). Without keeping an eye on the number of edits being made, the HTER score could be falsely inflated. Finding the minimal number of edits is, however, an ideal; there is no benchmark to compare to. And the task is difficult, with clear differences among editors in their ability to find a minimal set of edits (as discussed at the end of Sections 5.4.3.1.1 and 5.4.3.1.2). Therefore, to increase repeatability we create the composite version using the segments with the lowest HTER scores from the available second-pass post-edited outputs. In this context lower HTER scores are assumed to be better editing.

In the current phase of GALE, and for future phases, the post-editing protocol employs a streamlined two-stage process: in the first stage each system output is post edited by two independent teams of editors thus creating two post-edited versions, and

then in the second stage (exactly as in the previous third stage) we create and score a composite version.

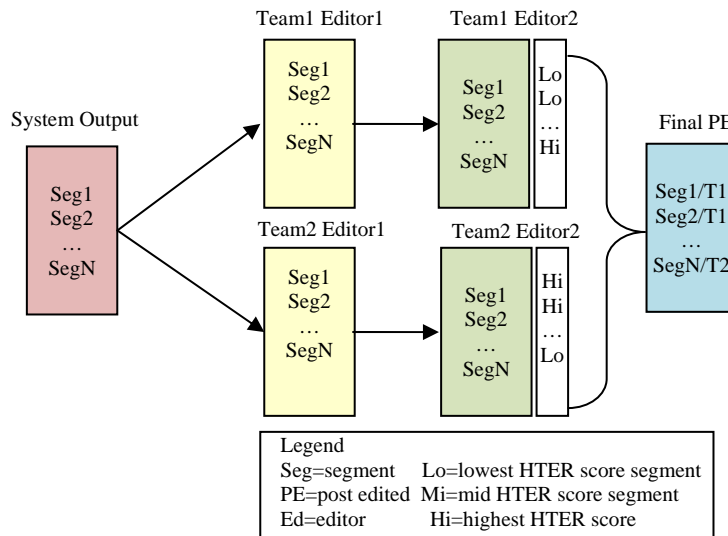


Figure 5.15: Diagram outlining the current post-editing protocol used in GALE

Each post editing team consists of two editors, one acting as the first-pass editor and the other serving as the second-pass reviewer checking the edits of the first pass editor. Together, the editors of a team produce one post-edited version. Thus, the two teams produce two post-edited versions. Unsurprisingly, with the variability that exists in human language, the resulting edits between two teams may at times vary, sometimes drastically. Ideally, five or more teams would be used for post editing each document and the differences between teams could be averaged for a better understanding of HTER scores. The practice of limiting to two teams has been dictated by both cost and the amount of data post edited for evaluation. HTER is calculated for each segment of each of the two versions. We select the version of each segment with the lower HTER, combine them into a composite document, and score the composite document.

Figure 5.15 summarizes the protocol used in GALE at the time of this writing.

5.4.3.3.5. Potential Problems and Attempted Solutions

There are two identified weaknesses in using the described post-editing protocols for MT evaluation over successive years, and both have to do with the human aspect of the task: (1) the lack of post-editors retention causing inter post-editor differences and (2) year-to-year intra post-editor differences.

Despite LDC's best efforts, it is not possible to retain the same set of post editors for duration of a five year program. During each phase of GALE, new editors are added to the team and some choose not to return. The quality of each post editor can range from those who are exceptional at solving the puzzle (finding the least number of edits required to modify MT output to contain the exact meaning of the reference translation using equally understandable English) to those who solve the puzzle in less than the

optimal fashion. Some editors may be aggressive in the amount of editing they perform to capture the meaning while others may be too lenient, taking too much editor liberty in deciding the meaning is equivalent. It is through the LDC's quality control process of spot checking editor performance that such trends may be found. LDC also compares each editing team's HTER scores to those of the alternative version. When a team is found to be producing consistently lower or higher HTER scores than the corresponding editing team, a flag is raised and a review of their work is conducted. On occasion, an editor has been counseled due to poor editing skills, but one has never been let go during the evaluation.

As the pool of editors change from evaluation to evaluation, "control documents" are used to measure the editing consistency across evaluations. This is accomplished by inserting original translations that were previously edited, in the preceding evaluation, into the current editing task. When the documents are edited by different individuals, this process allows the measurement of the average difference on a subset of documents between two evaluations. Major differences are not expected, but can result from changes in editing protocols or from changes in the performance of the editors. Major differences prompt investigation. Editor differences between two evaluations can help explain system performance changes between the two evaluations. If editors are found to be particularly picky one year, they can unknowingly mask system improvements.

5.4.3.4. Conclusion

This section described the evaluation metrology used for the evaluation of MT performance for the GALE program. HTER as defined is the primary metric for GALE evaluations. NIST and LDC jointly created the post editing guidelines. NIST has created the evaluation datasets. The Linguistic Data Consortium has coordinated and supervised the post editing work for the GALE evaluations.

5.4.4. GALE Machine Translation Metrology: Machine Translation Assessment

Authors: Mark Przybocki, Audrey Le, Stephanie Strassel, Calandra Tate Moore, and Gregor Leusch

5.4.4.1. Introduction

The DARPA GALE MT evaluations test a system's capability to accurately and fluently produce an English translation for language data that originated in one of two languages, either Arabic or Chinese. The official evaluation metric is HTER as described earlier in Sections 5.2.3 and 5.4.2. The evaluation paradigm requires local implementation, that is, GALE teams receive the evaluation source data, allowing processing to take place locally using their distributed translation system network. Due to the complexity and the costliness of post editing, each GALE Team submits a single system for evaluation representing the one-best translation. Contrastive systems and system components are not evaluated. In essence it is the quality of the returned hypothesized translations that is being evaluated.

There are two tasks evaluated in each GALE MT evaluation. These tasks differ only by the medium of source data being processed. We refer to the two tasks as the *Translation task*, when the data medium is text, and the *Transcription task*, when the data medium is speech or audio files.

5.4.4.2. Translation Task

The GALE MT *Translation task* requires systems to accurately and fluently translate foreign language text data into English. The text data has been drawn from various online news and web forums or blog sources. The evaluation data sets were designed to include some source providers that were represented in the allowable training data and some source providers who were not. The data encoding for both input and output files has remained UNICODE UTF-8 encoded data, throughout each phase of GALE.

Although the definition of the *Translation task* has enjoyed relative stability throughout each phase of GALE, there were two noticeable changes that occurred between Phases 1 and 2. The first allowed systems to use ground truth sentence segmentation, as defined from the original source data. In review of the phase 1 evaluation protocols, it was noticed that too many post-editing errors may have been improperly influenced by the automatic alignment process (Matusov 2005) that was used to align the system hypotheses to the reference translation. An automatic alignment process was necessary since the segmentations used for each team's system translations were created independently of each other and independently of the reference translation segmentation. Although post editors are trained to consider information in a document as a whole, allowing information to exist outside the current segment under focus, in practice many post editors were inconsistent with how they handled information in adjacent segments which could lead to over editing. By providing the true segmentation marks for the following phases these types of errors were and will be eliminated. The second modification was to reduce the size of the evaluated document. Rather than evaluating the contents of each full document, which ranged in size from about a hundred words to several hundred words, smaller cohesive subsets of the larger document were defined with each targeted to have about 200 words, thus allowing for more data points during analysis. This change was mainly due to a change in the program's targets. For Phase 1, the target was average accuracy. For the following phases, the target was set as the percentage of documents that exceeded a minimum accuracy target. To get a proper value for the percentage of documents, we increased the number of documents evaluated.

5.4.4.3. Transcription Task

The GALE MT *Transcription task* requires systems to accurately and fluently produce English transcripts from foreign language audio sources. Like the *Transcription task*, the evaluation data sets for Transcription were designed to include some source providers that were represented in the allowable training data and some source providers who were not. The source language audio files for the *Transcription task* were provided to the GALE teams as NIST à SPHERE (Garofolo 1994) formatted waveforms, most likely requiring systems to use an automatic speech recognizer (ASR) as the front end ideally optimized for use with a translation system. While both the ASR and MT system

components could be measured separately the focus of the GALE evaluations is not on the individual component performance but rather to evaluate the system performance as a whole.

As with the before mentioned *Translation task*, the *Transcription task* too had similar changes to begin allowing systems to use segmentation information, but this change for the *Transcription task* occurred after Phases 2 and was motivated out of the necessity to save time and to reduce the size of the evaluated audio file rather than any type of observed errors.

5.4.4.4. Data

Each succeeding GALE evaluation phase requires a new test set to be created for both evaluation source languages (Arabic and Chinese). The reason for the requirement for new test material arose because the performers used the results of previous evaluations for error analysis. This section defines the evaluation test sets used in the GALE phases. The evaluation test sets have been collected by the Linguistic Data Consortium (LDC). The actual selection of the data files to be used in evaluation has been a combined effort between NIST and the LDC, with final approval coming from DARPA.

The National Virtual Translation Center (NVTC) was responsible for generating the Gold Standard reference translations for the first evaluation (Phase 1) and the LDC has been responsible ever since.

Genres

At the highest level, the GALE evaluation data can be categorized into two distinct categories, structured data and unstructured data.

Structured data is defined as that which is cleaner, well prepared and most likely has been reviewed and edited before being published. There are two genres that represent structured data. The first is *newswire*. Newswire is text data that is typically a single coherent news story; examples can be viewed from any online news reporting agency. For the first phase, 3 Arabic and 2 Chinese sources were used in the evaluation. There was a concentrated effort to include more variety of source providers in the following phases (7 Arabic and 6 Chinese sources).

The second genre of structured data is *broadcast news* which contains smaller news clips extracted from a larger audio broadcast. As with text, broadcast news data represents clean audio broadcasts, usually scripted and this data has probably gone through multiple reviews before being broadcast. This data is typically void of conversational and spontaneous data. As with newswire, the first phase of GALE used 4 Arabic and 4 Chinese sources and this set was augmented to 17 Arabic and 14 Chinese sources.

Unstructured data genres, as the name implies, carry a more informal tone. In text data, the informality is found in the form of web data, blogs and forums where keyboarders are more likely to write in fragments, with typing errors, and using unconventional phrases and abbreviations. We refer to the unstructured text genre simply as *web data*. For audio data the informality comes in the form of call-in talk shows and

live reports from the field, including unscripted interviews. This genre is referred to as *broadcast conversations*. For Phase 1 UT, only 2 Arabic sources and one Chinese source of Newsgroups were used and for UA, 6 Arabic and 3 Chinese sources. This was augmented in following phases to 7 Arabic Newsgroups, 12 Weblogs, 2 Chinese Newsgroups and 9 Weblogs for UT. For UA, 24 Arabic sources and 14 Chinese sources were used in later phases.

We will use the following abbreviations throughout this chapter: NW to refer to the newswire genre, WB to refer to web data, BN for broadcast news, and BC for broadcast conversations.

Test Set Size

A reoccurring issue in defining an evaluation data set is that of determining “how much data is required for the test?” For GALE the “how much data” must be taken in consideration with the practical question of “how much will it cost” for the post editing of resulting system translations.

In Phase 1 the evaluation test set size was measured via source word counts with the goal to have approximately 10,000 source words represented in each of the four genres, for each source language. While this may seem relatively straight forward, a couple of assumptions were made. First, for Chinese text data, 1.5 native Chinese characters were estimated to correspond to one word. And since the selection of evaluation data was from audio (before reference transcriptions were available) an estimation of the rate of speech in Arabic and Chinese BN and BC data was required. The rate of speech estimates used for the various audio data sets are listed in Table 5.7 The adjustment between Phases 1 and 2 was made after post evaluation analysis identified that much more data was post edited in phase 1 than anticipated.

Language/Genre	Estimated Rate of Speech words per hour	
	Phase 1	Later Phases
Arabic BN	6000	5700
Arabic BC	6000	5700
Chinese BN	10,000	11,000
Chinese BC	10,000	11,000

Table 5.7: Estimated rates of speech for GALE audio evaluation data based on calculations from lower quality training data

Following Phase 1, the evaluation test set size was increased by 50%, that is, the goal was for each genre to be represented with 15,000 source words (for each language) and the number of documents was increased as explained above. Table 5.8 lists the actual words per genre as measured by the source and reference word counts. The decision to limit documents to smaller snippets (cohesive topic segments) allowed for the inclusion of many more document samples.

The fluctuation in test set difficulty from year-to-year has been an issue that NIST has struggled with in many human language technology evaluations. In the DARPA EARS program, NIST began making use of a “progress” test set that was treated with special instructions so that it could be reused in subsequent evaluations, allowing system improvements to be better tracked over time. This scenario only works if the evaluation

data can remain completely blind from the systems and participants. In GALE evaluations this is not possible. The evaluation data is examined in excruciating detail post evaluation. Post editors completely probe the data, and participants are encouraged to perform error analysis in order to improve their systems. Given these constraints, the program explored two alternatives for dealing with year-to-year test set difficulty differences. The first involved controlling the evaluation test set selection process, and the second made use of a few control documents allowing year-to-year test set calibration. These two ideas are further explained below.

Language/Genre	Phase 1				Later Phases			
	SRC Words	REF words	Doc Count	AVG Doc length	SRC words	REF words	Doc count	AVG Doc Length
Arabic BN	12k	15.5k	24	636	14k	20k	69	218
Arabic BC	11k	15k	13	1130	14k	20k	60	250
Arabic NW	10k	14k	51	182	14k	18k	68	220
Arabic WB	10k	12.5k	29	309	14k	19k	69	219
Chinese BN	13k	14.5k	18	777	16k	17k	70	215
Chinese BC	14k	13k	11	1126	15k	15k	53	283
Chinese NW	10k	10.5k	36	240	16k	17k	74	201
Chinese WB	10k	10.3k	19	485	15k	17k	66	227

Table 5.8: Word counts for phase 1 and phase 2 evaluation test sets. Source words are counts of native word tokens and reference words are counts of the words in the English translation.

Defining the Evaluation Epoch

Test set creation begins with selection of the evaluation (and optionally, a corresponding development test) epoch. Several factors are considered when choosing the evaluation epoch. First, the epoch should occur during a period of relative stability for LDC's ongoing data collection. For instance, LDC's broadcast collection system is taken offline once or twice a year to perform software upgrades and ongoing system maintenance, and an evaluation epoch that coincides with one of these maintenance windows would be subject to recording gaps. Similarly, programming schedules on the broadcast networks that LDC subscribes to are subject to variation during the time change between standard to daylight savings time, and so recording schedules may be unreliable for the few days surrounding this transition. Programming schedule disruptions are also common during holiday periods, for instance Chinese New Year and Ramadan. A second consideration in selecting the evaluation epoch is topical content in the news. While GALE evaluation data is intended to be representative of the linguistic challenges that exist in real world data, it happens occasionally that a single topic will dominate news coverage for days or weeks on end, and care must be taken to avoid selecting an epoch in which one single topic will predominate. For instance, the earthquake that occurred in the Sichuan region of China in May 2008 completely dominated Chinese news reports for several days on end, disrupting normal programming considerably.

Consideration must also be given to what data sets have already been "exposed" as training data (within GALE, or as part of other technology evaluation programs that GALE sites may have been part of); evaluation data should be blind and so previously-

exposed epochs and possibly sources are avoided. Finally, an evaluation epoch must be selected from early enough in the current phase to ensure adequate time to prepare Gold Standard references before system evaluation takes place.

At minimum, GALE evaluation epochs are one month in duration. This is necessary to ensure sufficient data volume and variety to support targeted selection. If a development test (devtest) set is to be defined, it typically comes from a time period directly preceding the evaluation epoch. Typically LDC and NIST collaborate to define an appropriate evaluation epoch, and the plan is endorsed or refined by the GALE data committee.

Evaluation Candidate Pool and Manual Selection

Once an evaluation epoch has been defined, LDC creates the evaluation candidate pool. The candidate pool is very large, typically hundreds of times larger than the final evaluation set. It consists of source data for all four genres that were collected during the evaluation epoch, processed to conform to the standard GALE source data format, and (where applicable) subject to manual or automatic auditing for quality and content (Chapter 1 in this book).

The candidate pool serves as input to the first stage of data selection. LDC annotators review the contents of the candidate pool to identify portions of documents or files that meet specific selection criteria. LDC has developed a customized user interface for doing data selection. (See Figure 5.16.) The interface uses a database backend to track annotation decisions. The process begins by dividing the candidate pool into randomized file lists. Each list includes up to 200 files, drawn from a single language and genre and either mixed or uniform with respect to sources/programs and epochs, depending on requirements. All files in the candidate pool appear on some list. At the start of a work session the annotator logs into LDC's Annotation Workflow System (AWS) ((Maeda 2009) and Chapter 1 in this book) and chooses Eval Data Selection from their assigned list of tasks. AWS then launches the data selection toolkit and pre-loads the next unassigned file list from the pool of available data. After the tool launches, the annotator sees the list of candidates to review, along with the first line of text in the file (where available) and the file size (in tokens or duration). The annotator clicks on each filename in turn, whereupon the tool displays the full text and/or audio for that file. Newswire candidates are presented for manual review as individual story units, with story boundaries pre-defined by the source data provider. Web text candidates are presented for review as threads, which consist of a single entry or post on a given topic, plus all the follow-up entries, posts, messages and/or comments responding to the original post. Broadcast candidates are presented for review as single audio files that correspond to a recording of one episode of a single program, typically 20, 30, or 60 minutes in duration, plus the corresponding automatic speech recognition (ASR) output for that recording (Chapter 1 in this book).

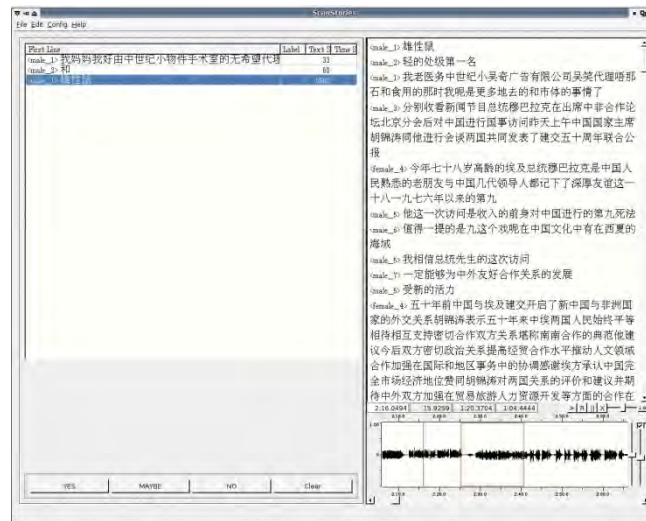


Figure 5.16: Audio and Corresponding Text Display for Selected Story



Figure 5.17: Completing Judgments for Selected Story

The annotator reads through the text or ASR output, and/or listens to the audio recording, and makes an initial YES/MAYBE/NO judgment about the suitability of a file for inclusion in the selection pool. The primary determination for suitability is based on topic content. Hard news about politics and current events, social issues, human interest stories, and opinions about current news are all examples of suitable topic content. This data may be collected as editorials, interviews, call-ins, round table discussions or from simple news reports. There are many categories of topic content that are excluded.

Some examples include: re-broadcasts or re-prints, previews, sporting events, lists of any kind, weather or stock reports, commercials, movie or book reviews, chain letters, fake news, and anything that contains explicit sexual or obscene language. Neither of these lists is intended to be exhaustive of all suitable and non-suitable topic content defined for use in GALE.

While collected source data varies in length, the GALE test set consists of smaller snippets - typically, segments under 200 words in length. (A snippet is defined as a single cohesive story or article contained in a much larger publication.) Because of this requirement, LDC annotators typically must designate one or more subsections of a given candidate file for selection. The toolkit includes functionality for marking off one or more segments (snippets) within an audio or text. When a segment is selected, the tool automatically updates the file inventory and file size (token or duration) information.

After the annotator labels a file or segment as YES or MAYBE, the tool launches a decision panel (see Figure 5.17) and the annotator records the following judgments:

- Genre: NW, WB, BN, BC, Other (with text box)
- Topic Category:
 1. Local or regional news – general
 2. National or international news – general
 3. Editorial or opinion
 4. Personal anecdote
 5. Human interest story
 6. Medical
 7. Legal
 8. Other
- Synopsis: 1-2 sentence description of content, in English
- Document difficulty rating:¹⁷ Less than 2; 2; 3; 3+ (See description of ILR ratings in Section 5.3.3.)
- Language/Dialect issues: text box
- Other comments: text box

After making the relevant judgments, the annotator moves on to the next file (or segment of the same file) for review.

At the close of the work session, the annotator is asked by AWS whether the file is complete or still in progress. If the file is labeled complete, AWS will allow the annotator to request another file for review or to log out. If the file is labeled in progress, AWS will launch the same file in the selection tool the next time the annotator logs in for a work session.

¹⁷ LDC annotators assign preliminary ratings using the Interagency Language Roundtable (ILR) system. Although there are no hard and fast requirements regarding document difficulty, the selection process primarily targets level 2 and level 3 documents. See http://www.dliflc.edu/academics/academic_affairs/dli_catalog/reading.htm

First Round Automatic Selection

Once manual selection on the candidate pool is complete, LDC annotators create quick rich transcripts for selected audio snippets, and segment the selected text snippets into sentence units, resulting in preliminary source text references (Chapter 1 in this book). LDC then distributes to NIST the preliminary source text references along with the output of the data selection database consisting of the following fields:

- Snippet ID
- Language
- Genre
- Source
- preliminary ILR
- Publication date
- Topic category
- Topic description
- Language / Dialect issues
- Document section location {Begin, Middle, End}
- Document section word count (text)
- Document section duration (audio)

The next stage of data selection applies a series of automatic diagnostics to calculate log-perplexity and 3-gram hit-rate for selected snippets.

First Round of Down Selection

Automatic measures of n -gram statistics and perplexity as measured against the original Phase 1 training data were used as a first step in controlling for year-to-year test set differences.¹⁸ The evaluation candidate pool of potential test materials was sampled, removing candidate documents that were outside the Phase 1 test material statistics in either n -gram hit rates or log-perplexity. From the remaining set of documents, NIST semi-randomly selected a set that was three times the anticipated test set size such that the n -gram hit rate distribution and log-perplexity scores were similar to that of the phase 1 test sets.

The selected files (three times the intended test set) were returned to the LDC for further processing. Audio sources were processed with in-house speech-to-text systems to generate rough transcriptions. These transcripts and the source data from the text genres were processed by in-house MT systems to produce system translations. The MT output and the rough human translations were returned to NIST for the second round of down selection.

Second Round of Down Selection

Before Phase 2, the hope was that matching n -gram statistics and perplexity measures would be sufficient to produce stable, in terms of test set difficulty, succeeding year test

¹⁸ This down-selection process did not apply to phase 1 evaluation data.

sets.¹⁹ It was not. In Phase 2, post evaluation analysis indicated that the Chinese test set was slightly more difficult than the Phase 1 test set which confounded the ability to demonstrate technology progress. Demonstrating progress is of vital importance in any evaluation that has strict targets of performance that must be met for continuation in the program. The fact that there was a measurable difference in test set difficulty was not a surprise and underscored the fact that inherent differences in the data exist despite best efforts to eliminate those differences. Language is ever changing.

A second round of down selection was implemented in Phase 3. In another attempt to reduce year-to-year test set differences the test material was carefully selected such that its overall TER distribution matched that of a previous year's test set. The idea is that an MT system is likely to produce similar TER distributions on two corpora with the same overall difficulty, provided that the corpora share the same genre, and that the system is not trained on either corpus. The use of system technology to select evaluation material is not a practice that NIST endorses. The same weaknesses that are a part of the system(s) being used to select the data will not be properly probed by the new test. Overriding the NIST concern is the need to demonstrate progress.

Along with the rough human translations of all the data being considered for test material, the LDC provided NIST with three system translations for each document, each produced by a different MT system. A per document TER score was established by averaging the scores from each MT system. In a similar manner TER scores were generated for each of the preceding evaluation test sets. The phase 3 test set was selected so that the TER distribution of the data matches the desired distribution. For the Arabic test materials, Phase 3 data was to match the TER distributions obtained in Phase 2. For Chinese test materials, Phase 3 data were to match the TER distributions obtained in phase 1.

This second round of down selection reduced the number of files to be submitted to Quality Control (QC) for final reference translation to the actual evaluation test set plus a handful of additional files, held in reserve, in case a file or two is deemed inappropriate for evaluation during the QC process.

While this second round of down selection was not performed to perfection in Phase 3, it is believed that in future phases of GALE this step will help reduce (but not eliminate) year-to-year test set differences. Short of using a progress test set – data that is kept blind and reused in future evaluations – language data is varied enough that efforts to control levels of difficulty will prove at times to be ineffective.

Final Reference QC

With the final selected files in hand, the LDC performs translation and QC, the last step in identifying possible issues with the selected data before declaring the official evaluation test set as finalized. The complete set of reference translations with segmentation that matches the source files are provided to NIST for GALE MT evaluation, which involves HTER scoring of translations from the evaluation systems.

¹⁹ This down-selection process did not apply to either of the first two phases of GALE. It has since become part of the data selection protocol and is planned for all future phases.

As described in Sections 5.2.3 and 5.4.2, HTER is a measure of edit distance between a modified system output and a single reference translation. Acknowledging that there are multiple acceptable translations HTER is designed to capture the quantity of edits required to obtain the correct “meaning.” Sometimes the source data can be ambiguous as to the intended meaning. In such cases alternative translations are encoded in the reference translation allowing the post editors to use their best judgment as to which “meaning” the system is closest to capturing. The actual implementation of alternatives in translation is described above in Section 5.4.2.

The LDC has created documentation that defines the process used to create reference translations. These translation guidelines are available from the LDC website.²⁰

Control Documents

As stated in the preceding sections, despite best efforts, some level of difference will often exist between two test sets. The GALE program has made good use of *control documents* to assist in understanding test set differences. Control documents are used in two facets.

In the first situation, several documents from an evaluation are declared *sequestered* shortly after the evaluation period. Team interaction with sequestered data is limited, and no system development is permitted that makes use of information derived from the sequestered files. In the subsequent evaluation these sequestered documents will be reprocessed by the GALE teams along with the new evaluation test set. The data is post-edited and scored. While not a part of the official GLE evaluation, the scores obtained from the sequestered data alone are useful as a benchmark of improvement from the previous year. This does not represent a perfect solution since the remaining data from the previous evaluation are used to develop the new system and that data is drawn from the same epoch where similar language, topics, and names are used, which might create a bias.

In the second situation, control documents are used to measure year-to-year editor differences. See Section 5.4.3.3.5 for a description of how control documents are used to measure changes in editor behavior from one evaluation to the next.

5.4.4.5. Metrics

In addition to the official GALE MT evaluation metric of HTER, other automatic metrics that GALE teams might use in the development cycle are reported to the teams to assist in error analysis. The automated metrics reported by NIST include: BLEU (Section 5.2.2.2 [version 1.04] above), METEOR (Section 5.2.2.3 [version 0.6] above), and TER (Section 5.2.2.4 [version 0.7.25] above). NIST reports per-document HTER scores that are used by DARPA to determine the success of teams meeting DARPA GALE program’s performance targets.

²⁰ <http://projects.ldc.upenn.edu/gale/Translation/>

5.4.4.6. Evaluation Rules

Each evaluation is implemented according to predefined rules and restrictions that identify the evaluation protocols. NIST documents these protocols in an evaluation specification document. The reader is referred to the NIST GALE web site for access to these evaluation specification documents (evaluation plans).²¹ A few of the rules and protocols are listed here.

1. Teams are forbidden from using data created or published during the same time epoch that the evaluation data is chosen from.
2. Teams are forbidden from manually inspecting the evaluation data before their results are submitted to NIST for scoring.
3. Teams are required to share all self collected training data.
4. Teams are required to process the entire evaluation test set and return resulting translations to NIST by an agreed upon due date, usually allowing two weeks to process the text sources, and three weeks to process the audio sources.

5.4.4.7. Systems Evaluated

As identified earlier there are three GALE teams. Each team submits a single set of results for each of the two test sets (Arabic and Chinese). The team names are AGILE (led by BBN Technologies), ROSETTA (led by IBM), and NIGHTINGALE (led by SRI International).

5.4.4.8. MT Results

In this section we report the performance results obtained in each of the GALE phases. Section 5.6 focuses on the primary metric of HTER. Section 5.4.4.8.2 focuses on the automated metrics that were reported to the GALE teams. Section 5.4.4.8.3 focuses on the intermediate HTER scores calculated from the first pass and second pass post editing.

²¹ <http://www.nist.gov/speech/tests/gale/>

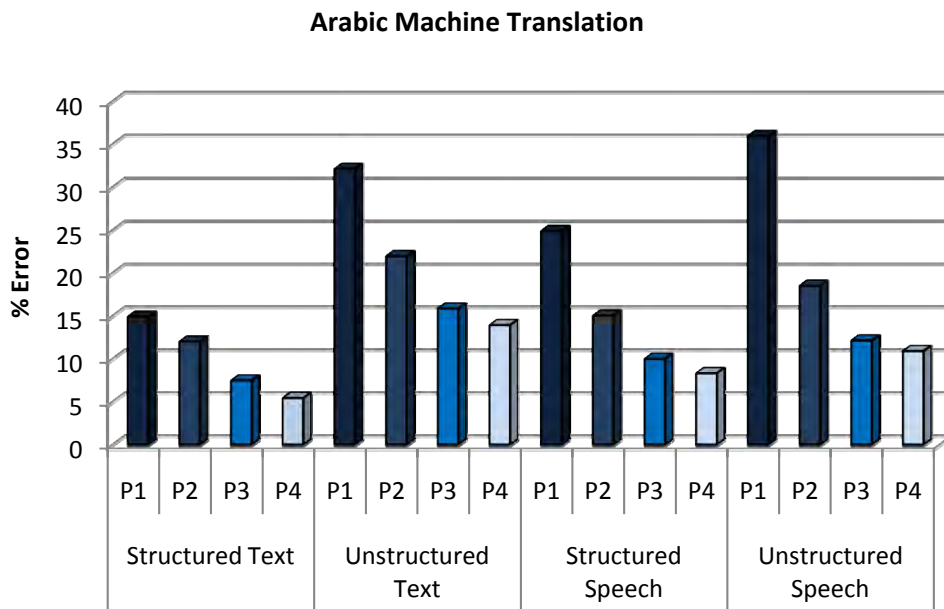


Figure 5.18: Average HTER for all genres of Arabic sources

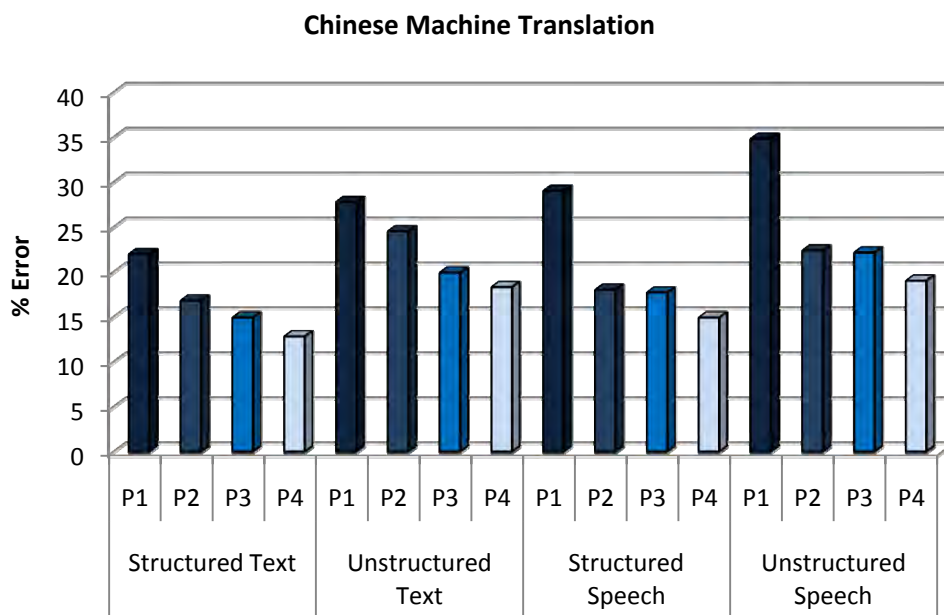


Figure 5.19: Average HTER for all genres of Chinese sources

5.4.4.8.1. HTER Results

The scores presented in this section represent the final HTER scores as calculated after completion of all post-editing. Scoring is divided into four genres: NW for newswire data, WB for web data, BN for broadcast news data, BC for broadcast conversation data. The results are shown in Figure 5.18 and Figure 5.19.

These averages were only used for the first year of GALE. For the following years, DARPA changed the metric. Instead of aiming for an average translation accuracy rate, the program set goals based on a given percentage of documents that exceeded a minimum accuracy. The targets for years two to five are shown in Table 5.9.

Language	Arabic				Chinese			
Genres	NW	WB	BN	BC	NW	WB	BN	BC
Phase 2	80/90	70/75	75/80	70/70	75/90	70/75	70/75	65/70
Phase 3	85/90	75/80	85/85	75/75	80/90	75/80	80/80	75/75
Phase 4	90/90	80/85	90/85	85/80	85/90	80/85	85/85	80/80
Phase 5	90/95	85/90	90/90	90/90	90/90	85/90	90/90	85/85

Table 5.9: GALE translation targets shown as accuracy/percent of documents

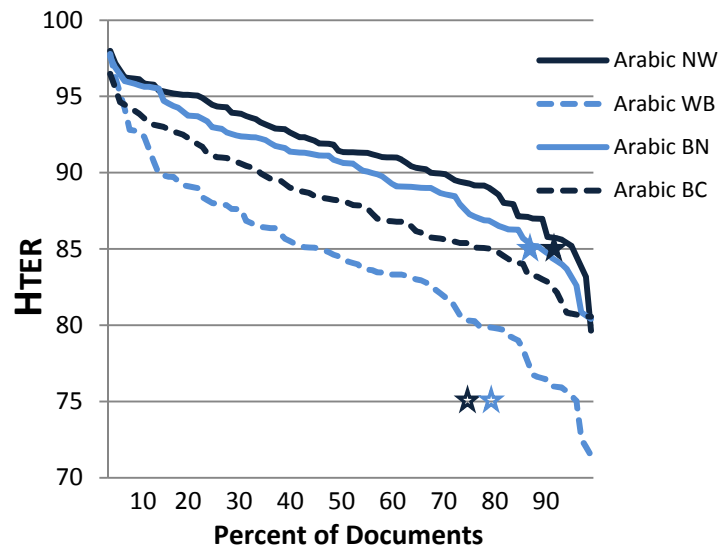


Figure 5.20: Arabic document accuracy plotted in descending order. Stars show the target for each Genre

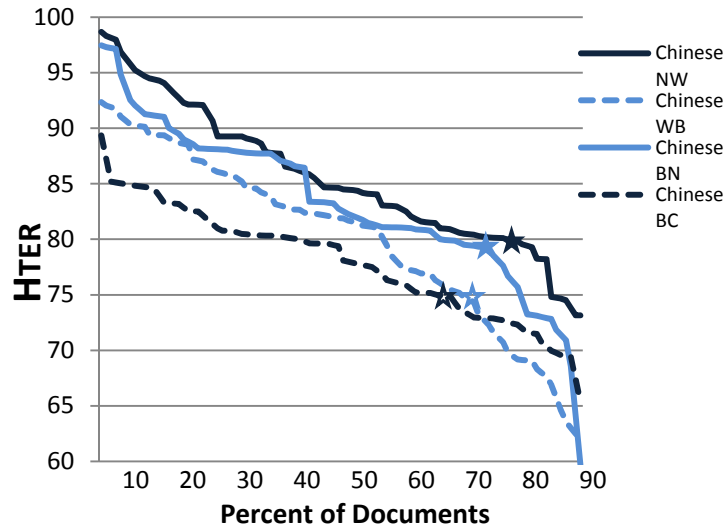


Figure 5.21: Chinese document accuracy plotted in descending order. Stars show the target for each Genre

Since the percentage of documents of a given accuracy was set as the GALE targets, the translation quality is best displayed in Figure 5.20 for Arabic and Figure 5.21 for Chinese. These figures display the document accuracies after they have been sorted. These are the best Phase 3 results. The figures show the Phase 3 targets as stars. All Arabic genre and three of four Chinese genres exceeded the targets. Only Chinese BN is slightly below target.

5.4.4.8.2. Automated Metric Results

Team	Genre	Average BLEU			Average METEOR			Average TER		
		P1	P2	P3	P1	P2	P3	P1	P2	P3
Agile	NW	0.23	0.33	0.4	0.55	0.62	0.67	0.55	0.45	0.4
	WB	0.11	0.22	0.27	0.40	0.53	0.57	0.69	0.58	0.52
	BN	0.16	0.27	0.35	0.48	0.58	0.64	0.64	0.53	0.43
	BC	0.14	0.23	0.28	0.44	0.55	0.57	0.68	0.57	0.49
Nightin gale	NW	0.23	0.31	0.32	0.55	0.61	0.61	0.56	0.50	0.48
	WB	0.10	0.20	0.21	0.40	0.52	0.52	0.70	0.64	0.60
	BN	0.16	0.23	0.28	0.48	0.56	0.58	0.69	0.61	0.51
	BC	0.12	0.23		0.42	0.53		0.75	0.64	
Rosetta	NW	0.24	0.29	0.38	0.56	0.61	0.66	0.55	0.47	0.41
	WB	0.13	0.19	0.25	0.42	0.50	0.55	0.68	0.58	0.53
	BN	0.17	0.28	0.35	0.47	0.59	0.64	0.70	0.60	0.50
	BC	0.13	0.20	0.27	0.42	0.50	0.57	0.70	0.60	0.50

Table 5.10: Arabic: Summary of Automated Metric Scores across Phases for BLEU, METEOR, TER (error measure). For each phase the average metric score is shown.

Several automated metrics were run over the submitted systems data compared against the single reference. Since these metrics do not make good use of embedded alternatives, only the first listed in the set of alternatives are used. The translation agencies are instructed to put the most common, or their best guess as to the intended meaning, first.

Team	Genre	Average BLEU			Average METEOR			Average TER		
		P1	P2	P3	P1	P2	P3	P1	P2	P3
Agile	NW	0.17	0.21	0.18	0.50	0.51	0.49	0.63	0.60	0.63
	WB	0.14	0.14	0.16	0.44	0.44	0.43	0.65	0.66	0.66
	BN	0.13	0.21	0.15	0.45	0.51	0.46	0.75	0.59	0.65
	BC	0.10	0.11	0.11	0.39	0.39	0.39	0.77	0.71	0.69
Nightingale	NW	0.14	0.17	0.16	0.48	0.49	0.47	0.67	0.64	0.65
	WB	0.11	0.13	0.13	0.41	0.42	0.42	0.69	0.68	0.67
	BN	0.12	0.19	0.14	0.43	0.49	0.45	0.73	0.62	0.66
	BC	0.08	0.11	0.11	0.37	0.37	0.39	0.78	0.70	0.68
Rosetta	NW	0.14	0.17		0.45	0.48		0.67	0.64	
	WB	0.10	0.12		0.39	0.40		0.71	0.68	
	BN	0.08	0.16		0.38	0.46		0.76	0.62	
	BC	0.06	0.10		0.33	0.36		0.79	0.70	

Table 5.11: Chinese: Summary of Automated Metric Scores across Phases for BLEU, METEOR, TER (error measure). For each phase the average metric score is shown.

Table 5.10 and Table 5.11 summarize the average scores for three popular automated metrics, for Arabic and Chinese, respectively. Please note that TER is an error measure so its polarity is different from the other two.

5.4.4.8.3. Correlation Study

Correlation results are generally used in MT evaluation to compare automatic and human evaluation of output from various machine translation systems. In this section, we present a correlation analysis of the results obtained from comparing each of the three engine outputs for the GALE data across evaluations, source language, and genre. We choose to use Spearman correlation as our measure for the strength of relationships between each of the variables of interest. Spearman rank correlation is a distribution-free rank statistic that tests the direction and strength of the relationship between two variables (Lehmann and D'Abbrera 1998). This method, based on ranking the two variables, makes fewer assumptions about the distribution of the values and measures monotone rather than linear covariation. We show the relationship between automated metrics (BLEU, NIST, TER, and METEOR) and both the human judgments of quality (as denoted by adequacy scores) and HTER (the official metric of GALE). Both automated metric scores and human judgments are calculated per segment and then correlated at that level. Adequacy scores amongst raters are averaged prior to performing correlations.

Analysis and Results

First, we computed correlation scores for the entire data set. Table 5.12 shows these results. The results presented are the absolute value correlations between the variables. This is done because as TER is an edit distance measure (smaller score means better), the true correlation between it and similarity metrics (larger score means better) has a reverse relationship. METEOR and HTER are found to correlate highest with adequacy as compared to the other metrics. As should be expected, TER correlate highest with HTER.

	HTER	BLEU	NIST	TER	METEOR
a)	0.622	0.508	0.577	0.517	0.624
b)	–	0.502	0.473	0.528	0.512

Table 5.12: Automated Metric Correlation with (a) Adequacy judgments and (b) HTER scores

	Language	HTER	BLEU	NIST	TER	METEOR
a)	Arabic	0.639	0.518	0.558	0.520	0.611
	Chinese	0.584	0.431	0.520	0.438	0.570
b)	Arabic	–	0.592	0.588	0.626	0.614
	Chinese	–	0.399	0.364	0.428	0.417

Table 5.13: Automated Metric Correlation with (a) Adequacy judgments and (b) HTER scores classified by Source Language.

We see in Table 5.13 that there is a change in the relationship between automated metrics and adequacy by source language. Within each language, HTER has higher correlation with adequacy scores. Across the board, scores are lower for Chinese than for Arabic. Similar findings have been confirmed in previous NIST MT evaluations.

Similar analysis shows a dramatic decrease in the magnitude of correlation within the unstructured data track of broadcast conversation across most metrics. With the exception of the broadcast news track, METEOR presents a stronger relationship with human judgment within genres.

Discussion of the Correlation Process

Correlation helps establish whether there is a relationship between machine translation output quality as determined by commonly used machine translation evaluation metrics and human induced adequacy or HTER judgments. What correlation does not provide is a correct sense of predictability of one for the other, i.e. it does not answer the question: Can automated metrics be used instead of human judgments? To correctly characterize a predictive relationship, we would need to extend beyond correlation and determine which automated metrics would best serve as predictors in distinguishing adequacy or HTER. Statistical modeling techniques would be useful in this direction.

Section Summary and Conclusion

We have shown various correlation results for several cross-classifications of the GALE data. In most instances, HTER and METEOR tend to be more closely related to human judgments of adequacy than the other automated evaluation metrics. Because HTER scores are just as time consuming to reproduce as adequacy metrics, it would be beneficial to find a metric that not only correlates well with adequacy but also HTER. While TER generally correlates better with HTER, it does not outperform other metrics in comparison to adequacy scores. Hence, METEOR seems to be the best automated metric in terms of assessing both adequacy and HTER. Though it does not correlate well to HTER, BLEU scoring remains the standard method for most internal development because of its low cost and rapid turnaround time.

5.5 Use of Evaluation for Optimization

5.5.1. Effect of MT Evaluation Measures on Optimization

Authors: Arne Mauser, Saša Hasan, Alok Parlikar, and Stephan Vogel

5.5.1.1. Introduction

Och (2003) suggested a procedure to automate research in MT with the help of a numerical optimization technique that works with an error surface on the MT output as defined by an automatic evaluation metric such as BLEU or TER. The error surface as defined in the space of model weights is known to be very bumpy. It has several sharp hills and valleys, and many plateaus. Numerical optimization can only guarantee a local optimum in this space. The final optimal value depends on where we start in this space, and also how the optimization algorithm searches the space. It has been shown that using random restarts in the weight space during search typically gives better optima (Moore and Quirk 2008). However, the search still remains a complex problem, which leads to a number of questions.

- How does selection of the tuning set impact the performance on unseen test data?
- How does optimization on one evaluation metric affect the performance on other metrics?
- Do different numerical optimization algorithms lead to different optima? How robust are these methods to the initial search point?
- Should all training samples be given the same importance or should, for example, optimization be biased to improve bad translations more?

Also note that any optimum found is still determined by an automatic metric, which may or may not reflect how humans actually perceive the optimized output.

In the following sections, we will explore these questions in a number of different experiments. Before doing so, we will first take a quick look at how MT systems are typically composed of a log-linear combination of models. We will also review the process of automatic optimization, and look at different numerical optimization techniques.

5.5.1.2. Minimum Error Rate Training

Minimum error rate training (MERT) is essentially a method for automatically optimizing the weights for each of our models (see Chapter 2) to achieve optimal performance. This is typically done in three steps: (i) choose an automatic evaluation metric that can judge how good or bad a particular translation output is; (ii) define an error function on the entire MT output with the help of this metric; and (iii) use a numerical optimization method to minimize this error function.

MERT can deal with several types of output formats from an MT system. The most common strategy is to use an n-best list of hypotheses. Given a vector of model weights, we can re-rank the hypotheses for each sentence in the n-best list and establish a new

first-best translation for those weights. Thus, the error function is defined by the metric error on the new first-best hypothesis for a given weight vector. Some MT systems can output an entire lattice of translations instead of just the n-best hypotheses. Lattices are compact representations of several hypotheses, so they contain more alternatives than what we would have in an n-best list. However, an error function on a lattice could be defined in a similar manner: for each sentence, choose the best hypothesis from the lattice, and the metric then defines the corresponding error.

The error function thus defined has several properties. First, it is not a convex function; it has many local optima. Secondly, the function is not necessarily differentiable or even continuous in the weight space, and a gradient function on the error surface cannot be analytically defined. Two widely used optimization techniques, which are applicable under such conditions, are the Downhill Simplex method and the Powell method.

5.5.1.3. Downhill Simplex

The Downhill Simplex method (Nelder 1965) is a commonly used nonlinear optimization algorithm. Given an N-dimensional space to search in, the method uses an N+1 dimensional polyhedron, called the simplex. The error is evaluated on each vertex of the simplex. The vertex with the highest error is discarded and a new test position is generated based on the behavior of the error on the different vertices. To avoid getting stuck in a local optimum and to achieve convergence, the downhill simplex algorithm includes a shrinking of the simplex.

5.5.1.4. Powell

The Powell method (Powell 1964) is another commonly used optimization method. This method uses a set of directions in the search space. Starting from an initial guess, the method moves along one dimension until a minimum is reached, then continues moving along the next direction until it finds the minimum, and thus cycles through the set of directions until the error function is minimized. It then modifies the set of directions, and repeats the process, until convergence is achieved.

5.5.1.5. Experiments

We conducted our experiments on the NIST MT Chinese-to-English task. Details are provided below regarding data characteristics, generalization to other data sets and metrics, and comparison of optimization techniques.

Data

The international evaluation held by the US National Institute of Standards and Technology (NIST) is focused on news translation of Arabic and Chinese to English. The bilingual training is provided by the Linguistic Data Consortium (LDC) and consists of newswire and news magazine translations, UN documents and parliamentary proceedings. In total, we have about 8 million sentence pairs or 250 million running words.

The evaluation corpus from 2002 is used as main development set. In most experiments we optimize the system weights on this corpus. The years 2003 to 2005 serve as test sets unless stated otherwise. Each corpus consists of about 600 to 1000 sentences and has 4 reference translations. All measures are computed disregarding the case of test. Evaluation of the corpus from the years 2003 to 2005 shows a degradation of 1.8 to 3 points in the BLEU score and a corresponding degradation in all other metrics evaluated (TER, WER, PER and NIST)

Generalization to other Metrics

Different MT evaluation metrics favor different features in the generated translations. For example TER typically favors shorter translations than BLEU. Therefore, optimization using one metric will not give optimal performance when evaluating using other metrics (Och 2003). However, if our MT system improve using one or two metrics, but degrades evaluating with other metrics, we can hardly feel confident that the translation quality really improved. Preferably, we would like to see an improvement on a number of different MT evaluation metrics.

	Evaluation					
Opt. on ↓	BLEU	TER	WER	PER	NIST	Avg. Len.
BLEU	35.9	56.7	63.1	39.7	9.63	31.8
TER	34.6	55.7	62.0	40.4	9.41	29.2
WER	33.2	55.5	61.0	41.8	8.93	27.4
PER	35.1	57.3	64.5	39.8	9.59	31.9
NIST	35.8	56.2	63.1	39.5	9.66	31.3
BLEU+ TER	35.4	55.8	62.2	39.8	9.56	30.2

Table 5.14: Error rates on the NIST 2005 corpus when optimizing an evaluating with different measures. TER, WER, PER: lower values are better, BLEU, NIST: higher values are better

In order to find a good overall criterion for system tuning, we examined the effect of optimizing on one measure and evaluating on all. The results for the NIST 2005 set are shown in Table 5.14. This table shows that the best results are obtained when evaluating with the same or similar metrics to the one used for training (BLEU, PER, and NIST vs. TER and WER), using non similar metrics shows some degradation. This grouping also shows a large difference in the sentence length. The word count for BLEU, NIST and PER shows longer hypotheses. TER and especially WER lead to rather short hypotheses.

Since the sentence length is rather different, we also tried to optimize on the sum of error-rate version of BLEU (100- BLEU) and TER. The result shows a good performance on most error measures, indicating that it could serve as a reasonable all-round criterion.

Comparing Simplex and Powell

Different optimization techniques will typically find different optima, even when starting from the same initial configuration. And starting from different initial configurations can lead to very different local optima. It is, therefore, advisable to use multiple random restarts (Moore and Quirk 2008). However, while finding a good local optimum is the primary goal, it is also important to find it fast. Some evaluation metrics

are computationally expensive. They can only be used in automatic tuning if the optimization procedure converges within a small number of steps, i.e., if only a small number of different hypotheses need to be evaluated.

We therefore studied how the two numerical optimization algorithms, Simplex and Powell, compare to each other in terms of speed of convergence and quality of the results. For this comparison we used the implementation of these algorithms from the SciPy toolkit. In this experiment, we used a 1500-best list of hypotheses on the MT05 test set for Chinese-English translation task. We separately used the TER and BLEU metrics to define the error function for optimization.

We chose ten different random points in the weight space as starting configurations and we ran both the Simplex and Powell methods to find the optimal value. We looked at what the resulting error value was, and how many different points in the weight space were evaluated before convergence. Ideally, we would like to have as low an error as possible, with as few as possible calls to the error evaluation. Note that error evaluation requires re-ranking an entire n-best list, and depending on the number of sentences in our test set and the size of the list, this operation can add significantly to the overall processing time. The comparison of the number of re-ranker calls required is shown in Figure 5.22. The corresponding BLEU scores achieved at the end of optimization starting from those random points is shown in Figure 5.23.

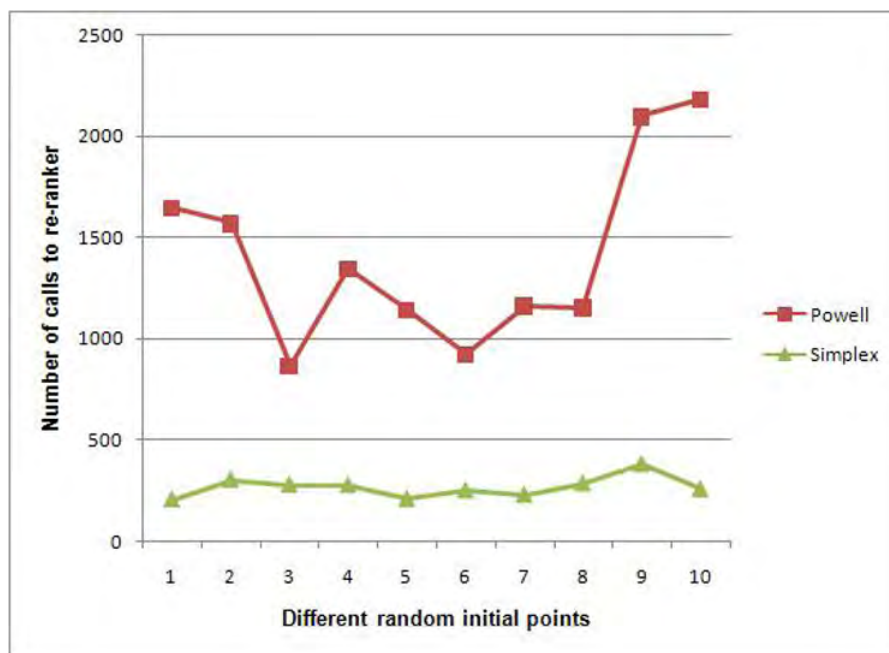


Figure 5.22: Comparison Powell vs. Simplex: Number of calls to re-ranker for different random starts.

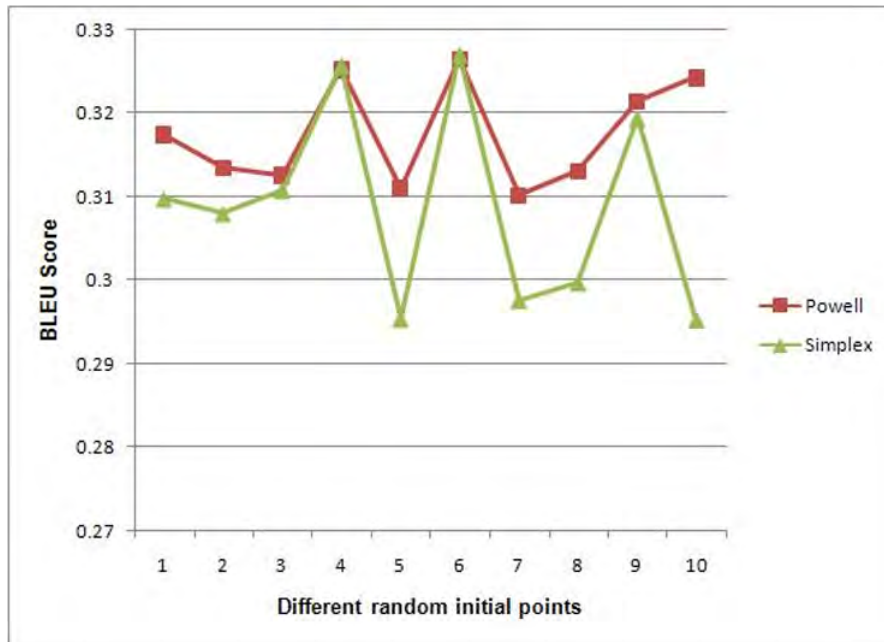


Figure 5.23: Comparison Powell vs. Simplex: Bleu scores for different random starts.

Although these figures are based on error function defined by the BLEU metric, a similar observation is made upon using TER. We observe that Powell's method consistently made many more function evaluations than simplex, and usually also ended up with a lower error than simplex method. We should note, however, that the difference between the Powell and Simplex methods, in terms of best BLEU scores, is rather small (0.008 average), and Simplex is much faster. I.e., we can run Simplex with more random restarts.

5.5.1.6. Document Level Tuning

MT systems are generally optimized towards the corpus level score of a dev-set. We have run some initial experiments to see what happens when we tune towards a collection of documents, each of which is weighted differently. The idea here is to investigate whether we can improve on hard-to-translate documents, by giving them more weight during optimization.

In our setup, we specify how many classes of documents we would like to have and we specify a weight for each class. After one round of decoding, we rank documents based on the BLEU scores of their top-best translations. We then chunk the ranked list of documents into stipulated classes. Based on what weight a document has, the metric features (n -gram counts in case of BLEU) for every sentence in the document are scaled up or down in the n -best list. MERT is run on this scaled n -best list, to find model weights that are then used in the next decoding iteration. Note that documents are re-ranked and reclassified in every MERT iteration, so they can have different weights every time.

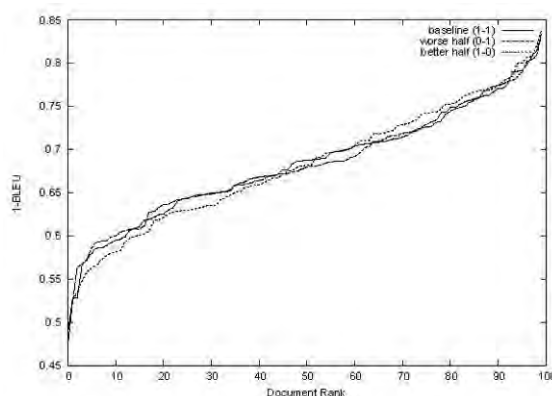


Figure 5.24: Document Scores on Dev Set

We ran initial experiments to understand how this document level tuning behaves. In one setup, we set the class weights to ‘1,0’, which means only the top 50% best translated documents participate in the optimization. In the other experiment, we set the weights to ‘0,1’, so only the worse half of the corpus participates in the optimization. Our baseline system uses all the documents, which is equivalent to having weights ‘1,1’. The dev-set used here is the NIST-MT05 set for Chinese-English translation. After four iterations of decode-MERT, we plotted the ranked errors of documents in each of the setups. We see that the error curve tilts in favor of the half that was used in tuning. Figure 5.24 shows this trend.

The scores of these three systems on an unseen test set, part of the DEV-07 set containing 588 sentences with one reference each. Tuning towards bad documents did not change the scores, but tuning to just the better half of the documents had a deleterious impact on both average scores, and the tail.

From these preliminary experiments, our hypothesis is that including poorly performing documents from the tuning set in optimization is important. We believe that assigning appropriate weights to the documents based on their scores can lead us to overall improvements, and we are investigating this. We notice that between the different MERT iterations, documents jump classes. Having a rigid 0-1 boundary is not desirable, because having a largely different set of documents in MERT could make it brittle. Instead, we plan to use a smoother weight distribution - using more than just two classes, and assigning weights in steps.

5.5.1.7. Conclusions

In this section, we looked at the automatic optimization of MT systems. We went over the Powell and Simplex methods of numerical optimization. Our experiments showed that Powell is more robust to random variation in the starting point in our setup. We presented a comparison of how tuning towards particular evaluation metric affects the performance along other metrics. We observed that BLEU and TER are robust metrics for optimization, because of two reasons. First, improvement on a development set results in improvement on similar test sets, and second, the improvement is seen consistently along all metrics of evaluation. This is not the case for metrics such as WER and PER, and

hence using either BLEU, or TER, or an interpolation of these two metrics seems to be an overall good choice for tuning an MT system.

5.6 Searching for Better Automatic MT Metrics

5.6.1. Searching for Better Automatic MT Metrics: Introduction

Author: Alon Lavie

This section describes recent research work by members of the GALE community on developing advanced automatic evaluation metrics for machine translation. The primary goal behind most of these efforts is to develop metrics with high levels of correlation between automatic metric scores and human judgments of translation quality. The commonly used metrics to date, previously described in Section 5.2.2, while effective for ranking systems, do not consistently correlate highly with human judgments of translation quality at the granularity levels of individual segments and documents. These levels are of particular interest for developers of MT systems within GALE, since the target goals of the GALE evaluations are formulated around percentiles of documents that score higher than set thresholds, as calculated by HTER scores. The new metrics described in this section are designed to address this fundamental issue. In some cases, the metric developers explicitly aim to directly improve correlation levels with HTER. This would provide MT system developers with fully automatic metrics for system evaluation and parameter optimization that are more effective in maximizing system performance, as measured by HTER.

We review six metrics in this section. While the primary goals of these metrics are fundamentally similar, the means by which they seek to achieve these goals are quite different. There are, however, several consistent themes. Several of the metrics (METEOR, TERP, RTE) seek improvements by addressing the language variability issue, where the same meaning can be expressed in various ways using different words and constructs. To address this variability, these metrics employ methods for detecting the similarity between an MT-produced translation and a reference at the word level (synonyms) and/or the phrasal level (paraphrases). A second common theme to several of the metrics (SEPIA, EDPMEmpty, MULCH, RTE) is their focus on syntactic features, in addition to, or instead of, features at the word and n -gram level. The rationale behind this is that similarity in meaning between an MT-produced translation and a reference may be more accurately identified by syntactic components of the two sentences, rather than correspondences at the shallow lexical-level. A third strong emerging theme for several of the metrics (METEOR, TERP, RTE) is their use of Machine Learning techniques for parameter optimization in order to obtain maximum correlation with human judgments. These metrics combine several features and components, which can be assigned relative weights. The idea behind this optimization process is similar to the goal of using automatic MT metrics for optimizing the performance of MT systems. In this case, however, the parameters of the metric itself are optimized to correlate best with human judgments on sets of data. Interestingly, this results in a two-stage optimization chain,

where an automatic metric is first optimized against human judgment data, and the resulting metric is then used to optimize MT systems. This gets around the difficult problem of optimizing MT systems directly against human judgments, which are difficult and expensive to obtain in quantities that would be required for direct optimization of the MT systems.

In the sub-sections below, we describe six different recently proposed automatic MT evaluation metrics. The descriptions include some initial assessments of improvements provided by the metrics in comparison with previous state-of-the-art metrics. All but one (MULCH) of these six metrics participated in the NIST MetricsMATR evaluation – an open comparative evaluation of automatic metrics for MT, which was organized by NIST in late 2008. Detailed contrastive results analyzing the performance of these and other metrics can be found on the NIST website.²²

5.6.2. Metric for Evaluation of Translation with Explicit Ordering (METEOR)

Authors: Alon Lavie, Abhaya Agarwal, and Michael Denkowski

The METEOR metric (Lavie 2004; Banerjee 2005) was originally developed in 2004, and the commonly used version of METEOR, version 0.7, tuned to adequacy and fluency, was already briefly described in Section 5.2.2.3. Here we describe some recent work on improving the metric and its correlation with human judgments. Much of the recent work on METEOR has focused on the tuning and optimization of the three adjustable parameters within the metric. This involved experimenting with optimizing the parameters on different data sets and to several different formulations of human judgments. The subsection below recaps the parameter optimization process used and presents the results of these optimization experiments.

5.6.2.1. Optimizing Metric Parameters

The original version of METEOR (Banerjee 2005) defines instantiated values for three parameters in the metric: one for controlling the relative weight of precision and recall in computing the Fmean score (α); one governing the shape of the penalty as a function of fragmentation (β) and one for the relative weight assigned to the fragmentation penalty (γ). For details, see the METEOR description in Section 5.2.2.3. In the original version of METEOR, these parameters were instantiated with the values $\alpha=0.9$, $\beta=3.0$ and $\gamma=0.5$, based on early data experimentation.

Over the past two years, in the context of the contrastive evaluations of automatic MT evaluation metrics conducted by the WMT-07, WMT-08, and the NIST MetricsMATR workshops, a thorough investigation was conducted aimed at tuning these parameters based on several available data sets, with the goal of finding parameter settings that maximize correlation with human judgments. Human judgments traditionally come in the form of “adequacy” and “fluency” quantitative scores. More recently, a new

²² <http://www.itl.nist.gov/iad/mig/tests/metricsmatr/>

form of human judgments in the form of binary and n-ary human rankings of translation output have also become available. The experiments conducted looked at optimizing parameters for each of these human judgment types separately, as well as optimizing parameters for the sum of adequacy and fluency. Parameter adaptation is also an issue in instances of METEOR that have been created for other languages, since parameters that were optimized to maximize correlation with human judgments for English would not necessarily be optimal for other languages.

In all of these optimization experiments, adequacy and fluency parameters were tuned by means of “hill climbing” and ranking parameters were turned by means of exhaustive grid search to find the parameters that achieve maximum correlation with human judgments on the training set. Pearson’s correlation coefficient was used as the measure of correlation. A “leave one out” training procedure was used in order to avoid over-fitting. Whenever training data segments from n MT systems was available for a particular language, parameters were trained n times, leaving the data from one system out in each training, and pooling the segments from all other systems. The final parameter values were calculated as the mean of the n sets of trained parameters that were obtained. When evaluating a set of parameters on test data, segment-level correlations with human judgments are computed for each of the systems in the test set and the mean over all systems is then reported.

5.6.2.2. Optimizing for Adequacy and Fluency

Parameters were trained to obtain maximum correlation with normalized adequacy and fluency judgments separately and also trained for maximal correlation with the sum of the two. Parameters for the sum of adequacy and fluency are used in the baseline versions of METEOR (versions 0.6 and 0.7), as these are the versions used in MT evaluations. To partially address human bias issues, the human judgments were *normalized*, which transforms the raw judgment scores so that they have similar distributions. The normalization method used is described by Blatz (2003). Multiple judgments were combined into a single number by taking their average. A detailed description of these experiments is presented by Lavie (2007).

	Adequacy	Fluency	Sum
α	0.82	0.78	0.81
β	1.0	0.75	0.83
γ	0.21	0.38	0.28

Table 5.15: Optimal Values of Tuned Parameters for Different Criteria for English

The resulting optimal parameter values on the training corpus are shown in Table 5.15. The optimal parameter values found are somewhat different than the previous settings of the metric parameters (lower values for all three parameters). The new parameters result in small, but noticeable improvements in correlation with human judgments on both training and testing data. Tests for statistical significance using bootstrap sampling indicate that the differences in correlation levels are all significant at the 95% level. An interesting observation is that precision receives slightly more “weight” when optimizing correlation with fluency judgments (versus when optimizing

correlation with adequacy). Recall, however, is still given more weight than precision. Another interesting observation is that the value of γ is higher for fluency optimization. Since the fragmentation penalty reflects word-ordering, which is closely related to fluency, these results are consistent with expectations. When optimizing correlation with the sum of adequacy and fluency, optimal values fall in between the values found for adequacy and fluency.

5.6.2.3. Optimizing for Rankings

Callison-Burch (2007) reported that the intercoder agreement on the task of assigning ranks to a given set of candidate hypotheses is much better than the intercoder agreement on the task of assigning a score to a hypothesis in isolation. Based on that finding, in WMT-08, only ranking judgments were collected from the human judges.

A new set of experiments was therefore conducted to re-optimize the METEOR parameters in order to maximize for correlation with ranking judgments. This required computing full rankings according to the metric and the humans and then computing a suitable correlation measure on those rankings. METEOR assigns a score between 0 and 1 to every translation hypothesis. This score can be converted to rankings trivially by assuming that a higher score indicates a better hypothesis. In development data, human rankings were available as binary judgments indicating the preferred hypothesis between a given pair. There were also cases where both the hypotheses in the pair were judged to be equal. These parameters are used in the METEOR-rank entry in some MT evaluations.

In experiments conducted for the WMT-08 evaluation, the binary ranking judgments were converted into full rankings of all translations of each source sentence. Spearman correlation was then calculated between these human rankings and METEOR scores rankings for each source sentence. The final correlation score is the average of the Spearman correlations for all individual sentences. An exhaustive grid search was performed in the feasible ranges of parameter values, looking for parameters that maximize the average Spearman correlation over the training data. To get a fair estimate of performance, three fold cross validation was performed on the development data. Final parameter values were chosen as the best performing set on the data pooled from all the folds. (see Agarwal 2008).

In further experiments for the NIST MetricsMATR evaluation, a slightly different optimization process was used. Binary human rankings were not converted into full rankings. Instead, the metric parameters were directly optimized for maximizing the number of correct binary human rankings across the entire training data set.

	A	β	γ	Original	Re-tuned
English	0.95	0.5	0.45	0.3813	0.4020
German	0.90	3.00	0.15	0.2166	0.2838
French	0.90	0.5	0.55	0.2992	0.3640
Spanish	0.90	0.5	0.55	0.2021	0.2186

Table 5.16: Optimal Values of Ranking-Tuned Parameters for Various Languages and Average Spearman Correlation with Human Rankings for METEOR on Development Data

The re-tuned parameter values from the WMT-08 experiments and the average Spearman correlations for various languages with original and re-tuned parameters are shown in Table 5.16. The parameters show that for optimal correlation with ranking judgments the “Precision/Recall” balance is weighed almost completely toward “Recall”, and this is consistent for all languages that we tested. Also, the re-tuned parameters for all the languages except German are quite similar. Comparing original and re-tuned parameters, we see significant improvements for all the languages. Gains are specially pronounced for German and French.

Overall, Meteor was one of the best performing metrics in the NIST MetricsMATR evaluation. Three versions of Meteor, differing only in their parameter settings, were evaluated. Meteor was one of the three top performing metrics in just about all conditions evaluating correlation with human judgments at the segment and document levels, for both adequacy and ranking forms of human judgments. Further details can be found on the NIST MetricsMATR website.

5.6.3. Translation Edit Rate Plus (TER-Plus)

Authors: Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz

While Translation Edit Rate (TER) (Snover 2004) has been shown to correlate well with human judgments of translation quality, it has several flaws, including the use of only a single reference translation and the measuring of similarity only by exact word matches between the hypothesis and the reference. These flaws are addressed through the use of Human-Mediated TER (HTER), but are not captured by the automatic metric. The handicap of using a single reference can be addressed by the construction of a lattice of reference translations, and such a technique has been used to combine the output of multiple translation systems (Rosti 2007). TERP does not utilize this methodology and instead focuses on addressing the exact matching flaw of TER.

TER-Plus (TERP) is an extension of TER that aligns words in the hypothesis and reference not only if they are exact matches but also if they are determined to be related morphologically or to be synonyms, as well as directly aligning multi-word phrases by considering possible paraphrases of the reference words.²³ Matching using stems and synonyms (Banerjee 2005) and using paraphrases (Zhou 2006; Kauchak 2006) have been shown to be beneficial for automatic MT evaluation. Paraphrases have also been shown to be useful in expanding the number of references used for parameter tuning (Madnani 2007; Madnani 2008) although they are not used directly in this fashion within TERP.

TERP uses all the edit operations of TER – Matches, Insertions, Deletions, Substitutions and Shifts, as well as, three new edit operations: Stem Matches, Synonym Matches and Phrase Substitutions. TERP identifies words in the hypothesis and reference that share the same stem using the Porter stemming algorithm (Porter 1980). Two words are determined to be synonyms if they share the same synonym set according to WordNet (Fellbaum 1998). Sequences of words in the reference are considered to be paraphrases of

²³ The name Terp doubles as the nickname – “Terp” — of the University of Maryland, College Park, mascot: the Diamondback Terrapin.

a sequence of words in the hypothesis if that phrase pair occurs in the TERP phrase table. Unlike TER, where all edits are given equal cost, the edit costs in TERP are optimized to maximize correlation with human judgments, and can be tuned to any particular type of human judgment.

5.6.3.1. Paraphrase Generation

TERP uses probabilistic phrasal substitutions to align phrases in the hypothesis with phrases in the reference. It does so by looking up – in a pre-computed phrase table – paraphrases of phrases in the reference and using its associated edit cost as the cost of performing a match against the hypothesis. The paraphrases used in TERP are extracted using the pivot-based method (Bannard 2005) with several additional filtering mechanisms to increase the precision. The corpus used for extraction was an Arabic-English newswire bitext containing a million sentences. Here are a few examples paraphrases that were actually used in a run of TERP:

controversy over → *polemic about*
by using power → *by force*
response → *reaction*

A probability for each paraphrase pair is computed as described by Bannard (2005). The phrase table for TERP contains 14,184,361 paraphrases. Paraphrases are only used if the reference side of the paraphrase occurs exactly in the reference translation, allowing us to filter the paraphrase phrase table according to a reference set if needed.

5.6.3.2. TERP Edit Cost Optimization

In total, TERP uses 11 parameters (Match, Insert, Deletion, Substitution, Stem, Synonym, Shift and Phase Substitution) out of which 4 represent the cost of phrasal substitutions. These costs are optimized to maximize correlation with human judgments. The match cost is held fixed at 0, so that only the 10 other parameters can vary during optimization. All edit costs, except for the phrasal substitution parameters, are also restricted to be positive. A simple hill-climbing search is used to optimize the edit costs by maximizing the segment-level correlation of human judgments with the TERP score.

With the exception of the phrase substitutions, the edit cost for all other edit operations is the same regardless of what the words in question are. That is, once the edit cost of an operation is determined via optimization, that operation costs the same no matter what words are under consideration. The cost of a phrase substitution, on the other hand, is a function of the probability of the paraphrase and the number of edits needed to align the two phrases according to TERP. Specifically, the cost of a phrase substitution between the reference phrase, p_1 and the hypothesis phrase p_2 is:

$$\begin{aligned} \text{cost}(p_1, p_2) = & w_1 + \\ & w_2 \text{edit}(p_1, p_2) \log(\text{Pr}(p_1, p_2)) + \\ & w_3 \text{edit}(p_1, p_2) \text{Pr}(p_1, p_2) + \\ & w_4 \text{edit}(p_1, p_2) \end{aligned} \quad (5.9)$$

editwhere $\text{edit}(p_1, p_2)$ is the number of edits according to TERP of aligning p_1 to p_2 and $\text{Pr}(p_1, p_2)$ is the probability of paraphrasing p_1 as p_2 , obtained from the TERP phrase table. Only paraphrases specified in the TERP phrase table are considered for phrase substitutions. In addition, the cost for a phrasal substitution is limited to values greater than or equal to 0, i.e., the substitution cost cannot be negative. The shifting constraints of TERP are also relaxed to allow shifting of paraphrases, stems, and synonyms.

5.6.3.3. TERP Alignment

In addition to providing a score indicating the quality of a translation, TERP also generates an alignment between the hypothesis and the reference, indicating which words are correct, incorrect, misplaced, or are close to the reference translation.

Reference		opponents	of	democratization	in	the muslim	arab	world	link	the			hamas							
Hyp After Shifts	I	S	S	T	D	P		S	S		I	I	I							
		and	advocates	to	democratize	the islamic	arab	region	between	the	republic	beats	"							
Reference		victory	to	the		election	gains	made	by	the	fundamentalist	movement	in	the	iranian					
Hyp After Shifts	P		I	I	P	D	Y	S		S	Y		D	D						
		won	the	"	and	electoral	gains	on	the	hard-line	trend	in								
Reference		elections		and	to	muslim	brotherhood	candidates	winning	five	seats	in	parliament	for	the	first	time	in	egypt	.
Hyp After Shifts		I	S	S					S											
		elections	iran	,	had	muslim	brotherhood	candidates	to	five	seats	in	parliament	for	the	first	time	in	egypt	.

Figure 5.25: Example of TERP HTML Alignment Output

Consider an example MT output from the NIST MetricsMATR MT-06 data set and the closest of the four reference translations. A portion of the HTML output of the alignment generated by TERP is shown in Figure 5.25. The alignment shown is the final alignment after all shifts are performed. Each word or phrase in the hypothesis is aligned to a word or phrase in the reference, with the symbol between the word or phrases indicating the type of edit: “I” for insertions, “D” for deletions, “S” for substitutions, “T” for stem matches, “Y” for synonym matches, and “P” for phrasal substitutions. The lack of a symbol indicates an exact match.

5.6.3.4. Correlation Results

	Optimization set			Test set			Optimization + Test		
	Seg	Doc	Sys	Seg	Doc	Sys	Seg	Doc	Sys
BLEU	0.623	0.867	0.952	0.563	0.852	0.948	0.603	0.861	0.954
METEOR	0.731	0.894	0.952	0.751	0.904	0.957	0.739	0.898	0.958
TER	-0.609	-0.864	-0.957	-0.607	-0.860	-0.959	-0.609	-0.863	-0.961
TERP	-0.782	-0.912	-0.996	-0.787	-0.918	-0.985	-0.784	-0.914	-0.994

Table 5.17: MATR MT06 Pearson Correlation Results.

A portion of the MT-06 data was annotated with adequacy judgments for the 2008 NIST MetricsMATR evaluation (Przybocki 2008) and distributed to participants as

development data. TERP was optimized to maximize segment level Pearson correlation with Adequacy on two-thirds of this data (the Optimization Set). In addition, we also show the actual Pearson correlation numbers on this optimization set and the remaining portion of the development data (the Test Set) in Table 5.17. For comparison, correlations from the standard TER, BLEU and METEOR are also shown. Those correlations that are statistically indistinguishable from the top metric (according to a 95% confidence interval) in each column are shown in **bold**. TERP had the highest correlation in each test case, and more importantly, showed significant improvement over the base TER evaluation metric.

5.6.4. Using Surface Span in Syntax-aware MT Evaluation (SEPIA)

Author: Nizar Habash

SEPIA is a newly developed MT evaluation metric that falls within the class of syntactically-aware evaluation metrics (Liu and Gildea 2005; Owczarzak 2007; Gimenez-Marquez 2007). The basic assumption of these metrics is that higher agreement in syntactic structure between MT output and Human reference translations reflects a higher degree of MT grammaticality and thus provides better correlation with human judgment than simple surface n -gram measures. An important failing in this assumption is that surface n -grams *do* capture a lot of syntactic information, especially for local dependencies. For instance, within the framework of dependency syntactic representation, 77% of all structural bigrams (parent-child links) have a surface span of less than 4 words and an absolute 47% of these are surface bigrams. Given this observation, SEPIA was designed with the goal of assigning bigger weight to structural bigrams with longer surface spans. Specifically, SEPIA uses a dependency representation but extends it to include surface span as a factor in the evaluation score. The dependency surface span is the surface distance between two words that are in a direct relationship in a dependency tree.

The rest of this sub-section describes the SEPIA metric and its variants, and summarizes how SEPIA performed in the recent NIST MetricsMATR evaluation (Habash 2008).

5.6.4.1. SEPIA

SEPIA evaluates a translation hypothesis segment (sentence) by computing a score based on a brevity-penalty-adjusted mean of multiple modified precision-based sub-scores. SEPIA uses two types of sub-scores: surface n -gram precision sub-scores (similar to BLEU (Papineni 2002)) and span-extended structural bigram precision sub-scores. We next discuss the latter type of sub-scores which are unique to SEPIA.

Span-Extended Structural Bigram Precision. A structural bigram (*SB*) is defined as a head word chain of size 2 (heads) in a dependency representation of the hypothesis/reference sentence. For example, in Figure 5.26, the edges linking the words *Among-crises*, *mentioned-Among* and *mentioned-dispute* represent *SBs*. An *SB* can simply be the parent-child word pair or it can include additional information such as the relation of child to parent (e.g., *Among-obj-crises*), the part-of-speech (POS) of both child and parent (e.g., *Among/IN-crises/NNS*), the relative order of the two (e.g., *Among- < -crises* or *mentioned->-Among*), or any combination of the above (e.g., *Among/IN- < -obj-crises/NNS*).

We define the surface span (*SS*) to be the absolute surface distance between parent and child in an *SB*. For the *SBs* *Among-crises* and *mentioned-Among*, the *SS* values are 5 and 12, respectively. Overall, in the tree in Figure 5.26, there are six *SBs* with *SS* of 1, two *SBs* with *SS* of 2, three *SBs* with *SS* of 3 and one *SB* each for *SS* values 4, 5, 10 and 12.

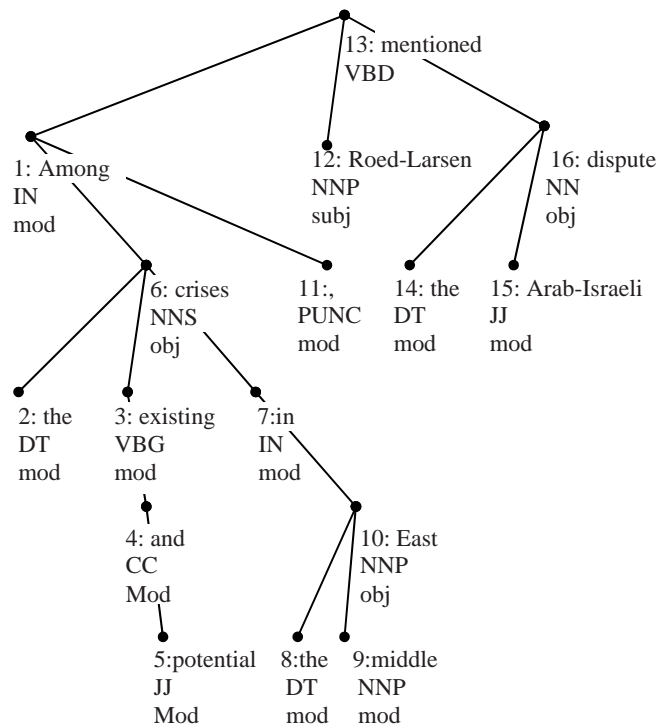


Figure 5.26: A dependency tree analysis for the sentence *Among the existing and potential crises in the Middle East, Roed-Larsen mentioned the Arab-Israeli dispute.*

For each unique *SS* value, n , associated with any *SB* in the hypothesis tree, we define SS_n as the count of all the *SBs* that have an *SS* value of n . We also define $SSclip_n$ as the count of all the hypotheses *SBs* (with *SS* value of n) that match reference *SBs*. However, if the number of matching hypothesis *SBs* exceeds the maximum seen in any reference tree, we use a partial count equal to (maximum # of reference *SBs* / # of hypothesis *SBs*)

in computing $SSclip_n$. This is our variant of *clipping*, used by other precision-based metrics (Papineni 2002) to minimize gaming.

Finally, we define the set $SPANS$ to contain all the unique SS values seen in the hypothesis tree.

Next, we describe two span-extended SB precision sub-scores, which vary in how they use the SS of an SB : SN_x and SPN .

First, the sub-score SN_x is computed as follows:

$$SN_x = \frac{\sum_{n \in SPANS} SSclip_n \times n^x}{\sum_{n \in SPANS} SS_n \times n^x} \quad (5.10)$$

SN_x is basically the span-weighted precision of hypothesis SB s matching reference SB s. The weighing is controlled through the power term x . The default value of x is 0, which assigns all SB s equal weight regardless of the SS value. A power term of 1 effectively multiplies the count of an SB by its SS value. A multiplier of 2 multiplies the count by the square of the SS value (and so on). This allows the user to give a bigger weight to the longer-distance matching spans.

Second, the sub-score SPN is computed as follows:

$$SPN = \frac{1}{|SPANS|} \sum_{n \in SPANS} \frac{SSclip_n}{SS_n} \quad (5.11)$$

SPN is basically the average of all SS -value-specific precision calculations. This scoring approach normalizes the frequency of SS values. This effectively gives more weight to the long-distance SB s because of the Zipfian distribution of SS s: shorter spans appear more frequently than longer spans.

Although the two scoring methods are different, they both give more weight to long-distance dependencies than to short-distance dependencies.

Sub-Score Combination. The segment-level SEPIA score is computed by taking the mean of any subset of the sub-scores described above, including both surface n -gram and SB sub-scores. Note that using the surface n -gram sub-scores alone is comparable to using BLEU. This serves as a robust back off in case no parse is found. The score is further adjusted by multiplying it with a brevity penalty factor as done in other precision-based metrics (Papineni 2002). Document-level scores are computed as a segment-length-weighted (in words) average of segment scores. Similarly, system-level scores are computed as a document-length-weighted (in segments) average of document scores. The set of all parameters in the SEPIA package is described by Habash (2008).

5.6.4.2. SEPIA's Performance in NIST MATR

For the NIST MetricsMATR evaluation, two SEPIA variants that are tuned on different data sets were submitted. SEPIA1 was tuned for MT-06 data, and uses surface unigram, bigram and trigram precision, in addition to the SPN subscore over (word, parent, relation, wordPOS, parentPOS and surface relative order). SEPIA2 was tuned for

TRANSTAC data, which was quite limited.²⁴ It only used surface unigrams and bigrams over morphologically de-inflected words. Both SEPIA variants employed length penalty. In terms of speed, SEPIA1 and SEPIA2 ranked 28th and 29th, respectively, out of the 38 NIST MetricsMATR metrics. Both took around 2 hours to finish (compared to BLEU taking less than 5 minutes but faster than metrics that took over a day). SEPIA1 was the best performer in two conditions: system-level correlation with multi-referenced pairwise and yes/no conditions. Sepia1 was in the top 5 performers in 12 cases out of 45. SEPIA1 always outperformed SEPIA2, which was in top 5 performers in 4 out of 45 conditions. SEPIA1 and SEPIA2 did better with multiple references than single references. They also did better at document and system level than sentence level: neither SEPIA1 nor SEPIA2 was in top 5 performers at sentence-level correlation.

5.6.5. Expected Dependency Pair Match (EDPM)

Authors: Jeremy Kahn, Mari Ostendorf, and Brian Roark

5.6.5.1. Introduction

In improving MT metrics, we try to model acceptable variation, whether by modeling word-choice (e.g., METEOR (Banerjee 2005)), by weighting adjacent matches more than non-local matches (e.g. GTM (Turian 2003)) or by modeling syntactic information (Liu 2005; Owczarzak 2007). In keeping to the syntactic approach, we consider a family of Dependency Pair Match (DPM) measures, which follows the labeled-dependency match version of SPARSEVAL (Roark 2006) and the **d/d_var** (Owczarzak 2007) measures. These approaches evaluate hypothesis-reference similarity with an F measure over fragments of a labeled dependency structure, which may be generated by a PCFG with deterministic head-finding (Liu 2005; Roark 2006 SParseval) or by extracting the semantic dependencies from an LFG parser (Cahill *et al.* 2004 and Owczarzak 2007). Our specific extension, Expected Dependency Pair Match (EDPM), leverages a publicly available PCFG parser, deterministic head-finding rules, word-level matching and weighted multiple parse alternatives for improved performance.

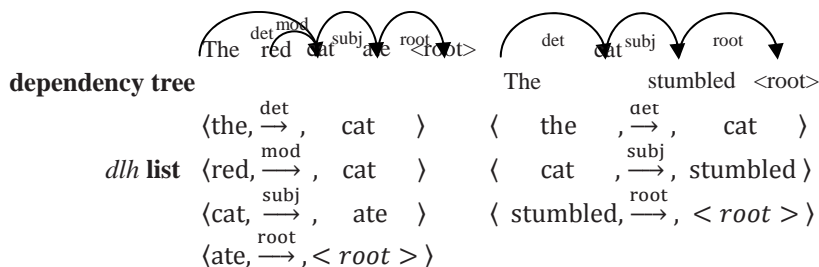
5.6.5.2. DPM Family of Metrics

DPM is defined as the F measure over bags-of-subtrees of the hypothesis translation dependency tree as compared to bags-of-subtrees of the reference translation dependency tree.

Different members of the DPM family of metrics may extract different subtrees. We denote the set of extracted tree-components with a trailing DPM_{dlh} extracts all $\langle \text{Depend}, \text{arc-Label}, \text{Head} \rangle$ subtree tuples, roughly equivalent to labeled SPARSEVAL (Roark 2006 Sparseval) and the (Owczarzak 2007) **d** measure. We also consider the $DPM_{1g,2g}$ extractions, which represent unigrams and bigrams, or the $DPM_{dl,lh}$, which extracts all the

²⁴ Spoken Language Communication and Translation System for Tactical Use (TRANSTAC). See <http://www.darpa.mil/IPTO/programs/transtac/transtac.asp>.

subtrees $\langle \text{Depend, arc-Label} \rangle$ and $\langle \text{arc-Label, Head} \rangle$ (roughly equivalent to the (Owczarzak 2007) **d_var** method).



Precision_{dlh} is $\frac{1}{3}$ and Recall_{dlh} is $\frac{1}{4}$, yielding $DPM_{dlh} = F = \frac{2}{7} = 0.286$

Figure 5.27: Example hypothesis and reference dependency trees and the *dlh* decomposition of each.

$\langle \text{Depend, arc-Label, Head} \rangle$

Since the dependency structures of the hypothesis and reference text are hidden, we also explore alternative dependency structures predicted by the parser, to cope with genuine ambiguity (in both hypothesis and reference) and to mitigate the effects of parser error. DPM is well-defined over the n-best list of dependency-structures: when $n > 1$ DPM uses the expectation of bags-of-subtrees rather than the bags-of-subtrees derived from the 1-best parse. In computing the expectation, we first “flatten” the parser probabilities such that $\tilde{p}(x) = \frac{p(x)^\gamma}{\sum_i p(i)^\gamma}$ (where γ is a free parameter) to account for over-confidence in the parser, normalizing so that the resulting probabilities sum to one over the n-best list.

In principle, the DPM family of measures may be implemented with any parser that generates a dependency graph (a single labeled arc for each word, pointing to its head-word). Our reference implementation uses a state-of-the-art PCFG parser (the first stage of Charniak-Johnson, (2005)) to generate a 50-best list of trees for each hypothesis and reference translation, using the parser’s default WSJ (Wall Street Journal) training parameters.²⁵ We use Magerman (1995) head-finding to construct dependency trees, using the Charniak parser’s head-finding rules, with three modifications: prepositional and complementizer phrases choose nominal and verbal heads respectively and auxiliary verbs are modifiers of main verbs (rather than the converse).

²⁵ <http://ssli.ee.washington.edu/people/jgk/dist/edpm/>

Arc-labels $d \xrightarrow{A/B} h$ are determined from constituent labels, where the arc label A/B between dependent d and its head h is composed of A (the lowest constituent headed by h and dominating d) and B (the highest constituent headed by d). This strategy is the one adopted in labeled-dependency SPARSEVAL, and it acts as an approximation of the rich semantics generated by Cahill *et al.* (2004), but with much less knowledge-engineering required. The A/B labels are not as descriptive as LFG semantics, but they have a similar resolution, e.g. the $\xrightarrow{S/NP}$ arc label usually represents a subject dependent of a sentential verb.

5.6.5.3. Experimental Validation

Experiments exploring correlation with fluency and adequacy judgments against the LDC Multiple Translation Chinese corpus Chapters 2 and 4 indicate that the best member of the DPM family uses the full 50-best parses produced by the system, with a “flattening” $\lambda = 0.25$. Using both the partial subtrees dl, lh and the string-only statistics $1g, 2g$ provides an optimal setting of:

$$EDPM = DPM_{1g, 2g, dl, lh, n = 50, \gamma = 0.25}$$

This configuration for EDPM has a correlation $r=0.24$ against the average fluency and adequacy judgment per-sentence over these corpora.

Measure s	All-Arabic	All-Chinese	All
TER	0.51	0.19	0.39
BLEU	-0.40	-0.19	-0.32
EDPM	-0.61	-0.25	-0.47

Table 5.18: Per-document Pearson’s r of Δs with ΔH_{ter} over various measures s , examined for each genre in the corpus, for each language in the corpus, and as a whole.

We also explore the EDPM variant’s utility on the task of predicting the human-targeted translation edit rate (HTER) on the (unsequestered) GALE evaluation results, and find that per-document differences across systems in EDPM ($\Delta EDPM$) are better correlated with changes in HTER (ΔH_{ter}) than $\Delta BLEU$ or ΔTER (Table 5.18).

Note that HTER uses a TER measure to calculate the post-editing work between the hypothesized translation and the human-targeted reference which could, in principle, bias HTER towards a TER measure. EDPM shares no such advantage. EDPM nevertheless has the best correlation of the three measures in both Arabic and Chinese, as well as over the entire corpus.

5.6.5.4. Future Work

Within the DPM framework, there are a number of directions for improving the quality of the scoring function, including increasing the number of parse alternatives, using different parsers, and exploring other (higher order) segmentations of the dependency tree. It would also be useful to assess the sensitivity of the score to parse quality, since very poor quality translations might be difficult to parse.

Alternatively, one might look at combining EDPM with other measures that attempt to account for allowable variation in word choice. Unlike many recently proposed evaluation methods, EDPM does *not* use word-substitution tables or tuned weights (beyond the λ free parameter described above), and yet substantially outperforms BLEU and TER as a predictor of changes to HTER. Perhaps further gains could be achieved by combining the different approaches.

5.6.6. Metric Using Large Chunks of Hierarchy (MULCH)

Authors: Seth Kulick and Mitch Marcus

5.6.6.1. Introduction

Our work aims to improve automatic MT evaluation by working with the syntactic structure of the reference sentence rather than treating it as a string of words. Our hypothesis is that this will allow a metric to make more sophisticated decisions about the relationship between the system output and the reference sentence, particularly in regard to such aspects as word movement and substitution. We are testing this hypothesis by adapting the METEOR system to use the treebanked representations of reference sentences that are available as part of the Ontonotes project. We refer to this new evaluation approach as MULCH (“Metric Using Large Chunks of Hierarchy”).

METEOR has two basic components. The first is a unigram matching between the system and reference sentences, where the matching is done by exact match and also looser matching using WordNet. The second is a “fragmentation penalty” to account for word order, attempting to capture such information as arguments in the wrong order.

We have decomposed the trees for the reference sentences into small tree fragments, representing core pieces of argument structure, with adjuncts and arguments separated. The two components of our adapted METEOR view the reference sentence as a collection of these small tree fragments, with the following impact:

- The unigram matching is able to differentiate between “function words” that can easily differ between a system output and reference translation, and those words that capture a core semantic notion. For example, the complementizer *that* might be missing from a subordinate clause, or the preposition *about* might be used instead of *on* in a prepositional clause. In such cases, by referring to the small tree fragments, we can exclude such function words from the recall calculation.
- METEOR’s fragmentation penalty is not able to distinguish between adjuncts and arguments, although they have very different word order possibilities. We have taken the obvious step, now that we have the trees, of completely rewriting the fragmentation penalty to require arguments to be in the correct order, while adjuncts are free to move.

5.6.6.2. Example

We present here an example sentence from the output of one of the GALE systems, from the Arabic Newswire section of the data. The reference translation for the sentence is (1), and the system output is (2).

- (1) In July 2005 Turkey had signed a protocol ...
- (2) Turkey signed in July 2005 a protocol ...

The unigram matching component of METEOR leaves *had* in the reference sentence unmatched, lowering the recall score. In addition, the “fragmentation penalty” lowers the score because the words *In July 2005 Turkey had signed* in the reference sentence do not map to a continuous sequence in the system output, due to the movement of *in July 2005*.

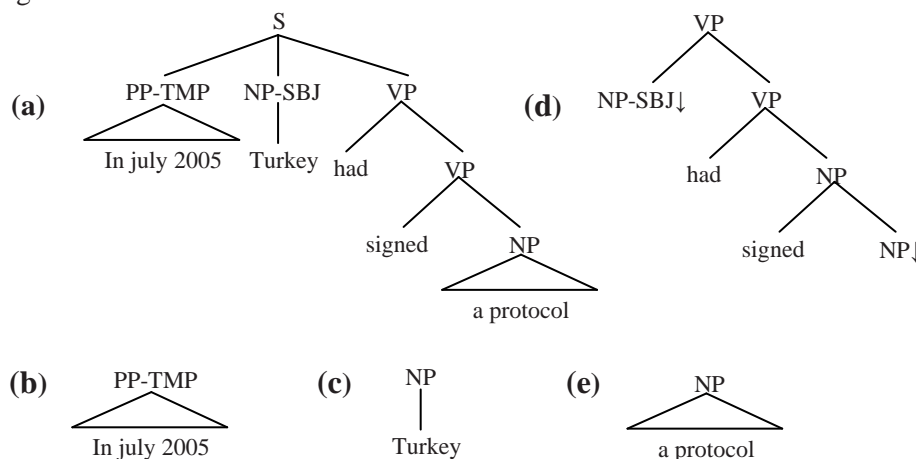
These are clearly not actual errors in the system output though, but rather simply an alternative translation, different from the reference sentence, but just as valid. However, since METEOR only works with words as strings with no linguistic information, it is unable to recognize this and compensate appropriately.

In our approach, we utilize the treebanked representation of the reference string, of which the relevant section is shown below (a). We decompose this tree into smaller tree fragments, associating each one with a lexical item.²⁶ This results in the four trees in (b),

²⁶ This is very much inspired by the long line of work in Tree Adjoining Grammar Joshi and Schabes (1997).

(c) (d) and (e), as well as a “derivation tree”, not shown here, which encodes how the tree fragments fit together to form the original tree.²⁷

As mentioned above, we can now improve the unigram matching F measure of METEOR by using these tree fragments as a small domain in which the matching can be made loosened. For example, we have encoded that if there is a match with the main verb of a tree, there is no penalty if an auxiliary that is present in the same tree does not map to a word in the system output. In this example, this means that since *signed* in the tree (d) is mapped to *signed* in the system output, there is no penalty for *had* not having a mapping.



Also, while we do not have space to show all the details here, these trees can be used to check that verbs and their arguments are in the same order in the system output as in the reference translation. We have completely rewritten the “fragmentation penalty” aspect of METEOR to take advantage of this new source of information. In this example, the order of *Turkey* and *a protocol* around *signed* is enforced, using the information in tree (d). However, the adjunct in *July 2005* is allowed to appear anywhere and so there is no penalty in our revised METEOR for *in July 2005* appearing in different locations in the reference and system output sentences.²⁸

It is worth noting that to some extent our use of the trees is able to simulate what the human annotator does in the edited reference sentences for the HTER evaluation. For example, for this particular sentence the edited reference sentence is edited:

(3) Turkey signed in July 2005 a protocol ...

²⁷ The ↓ indicates a node where another tree must be substituted into it, such as the tree for Turkey in (c) into the NP-SBJ node in (d).

²⁸ This is of course a simplification, since adjuncts do not have complete freedom of movement.

What the human annotator has done is adjust for the missing *had* and the movement of *in July 2005*, just as we have been able to do automatically by making use of the linguistic information encoded in the trees.

5.6.6.3. Evaluation

To test the modified METEOR, we calculated the following three scores for a team's output, for each sentence:

- HTER: using tercom on the system output and human-edited reference sentence.
- Original METEOR: using the system output and the (unedited) reference sentence.
- Modified METEOR: using the same data as for the previous, along with the treebanked representation of the reference sentence.

We then calculate the Pearson correlation between (HTER, original METEOR) and (HTER, modified METEOR).

Focusing for now on the Arabic Newswire part of the data, we obtain the scores for a development slice of the data, showing the increase in correlation from (HTER, original METEOR) to (HTER, modified METEOR):

AGILE:	0.66 → 0.69
ROSETTA:	0.65 → 0.66
NIGHTINGALE:	0.62 → 0.63

While there is some improvement, it is clearly not enough. Our hypothesis, based on some examination of the data, is that while our modified METEOR is able to improve the correlation with HTER when the system output and reference translation are relatively close, it has very little to work with if the system output and reference translation are far apart, since in that case the edited reference sentence used for HTER is so greatly different from the original reference that our modified METEOR is not able to simulate the construction of the edited reference sentence.

That is, we view the construction of the HTER sentence as consisting of four aspects:

- deletions/insertions: METEOR tracks this, while our modified version can improve this.
- synonyms: METEOR tracks this.
- word order movement: METEOR tracks this, while our modified version can improve this.
- severe paraphrases substantially altering the reference sentence. Neither METEOR nor our modified version can track this.

Our plan is to measure the distance between the original and edited references, in effect measuring “severe paraphrases”, and determine if there is a threshold of distance such that for distances less than that threshold our modified METEOR shows a substantial improvement in correlation compared with the original METEOR.

5.6.7. The Stanford RTE-based Metrics (RTE and RTE+MT)

Authors: Sebastian Pado, Michel Galley, and Christopher Manning

5.6.7.1. Introduction

The first generation of metrics developed for the automatic evaluation of machine translation output (such as BLEU (Papineni *et al.* 2002) and NIST (Doddington 2002)) were based on surface overlap, looking for identical word sequences between a system translation and one or more reference translations. However, recent work such as Callison-Burch *et al.* (2006) has identified serious problems with these metrics, such as unreliability at the level of individual segments, a bias towards statistical phrase-based MT systems, and (at least for some language pairs) low correlations between scores and human judgments. This casts a shadow on the frequent use of higher BLEU scores as a necessary and sufficient indicator of improvement in MT.

More recent metrics address these shortcomings by refining the matching problem, for example by phrasing it as an edit sequence (Ter, Snover *et al.* 2006), or by integrating synonymy information from WordNet (Meteor, Banerjee and Lavie 2005). Our proposal is to go one step further and determine the adequacy of a system hypothesis with respect to a reference translation completely on a semantic level that generalizes over the different kinds of variability of linguistic realization that can arise through syntactic rearrangements (such as active/passive diatheses or adverb placement) or lexical and paraphrastic variation. To do so, we exploit the considerable similarities between the MT evaluation task and the task of Recognition of Textual Entailment or RTE (Dagan *et al.* 2006). Given two short segments of text (premise and hypothesis), the recognition of textual entailment decides whether the hypothesis is entailed by the premise or not. Knowledge about the likelihood of textual entailment has been found to be beneficial for a range of applications, e.g., Word Sense Disambiguation or answer validation in Question Answering (Harabagiu and Hickl 2006).

Transferred to the domain of MT evaluation, our intuition is that a good translation should be equivalent in meaning to the reference translation, Hence the likelihood of mutual entailment between system output and reference should be high for good translations, and low for bad translations.

However, there are also evident differences between RTE and MT evaluation, both in terms of the data (entailment usually assumes that sentences are well-formed, which is often not true in MT), and in terms of the task (entailment is a “stricter”, and asymmetrical, task). Thus, we advocate a modular approach that predicts MT quality on the basis of entailment features, but can adjust the importance of these features for the new task. The resulting entailment-inspired approach has the potential of addressing the problems described in the first paragraph. First, entailment decisions are naturally made at the segment level. Second, textual entailment provides a framework for the integration

of different kinds of linguistic analysis techniques into MT evaluation. Finally, a deeper linguistic analysis can reduce biases towards particular system architectures.

5.6.7.2. Metric Implementation

The entailment features we use are computed by the Stanford RTE system (MacCartney *et al.* 2006). The input to the system is a pair of a premise and a hypothesis. We first perform a linguistic analysis whose result is a typed dependency graph. Next, an alignment is constructed between words of either sentence as the alignment that scores highest according to lexical similarity scores from about ten semantic resources and syntactic parallelism features. In the third phase, we construct features which model a range of syntactic, lexical, and semantic phenomena, such as factivity, polarity, monotonicity, matches and mismatches of named entities, and the quality of the alignment. The final score for each segment pair is computed by a regression model over the set of these entailment features. In contrast to the general entailment task, the features are computed for both directions to make a prediction for MT evaluation. This captures that system output and reference should entail each other.

We compare this system (RTE) against two individual MT metrics (BLEU, TER) and a regression model over different parameterizations of the BLEU, NIST, and TER metrics (MT). Finally, we combine the features of the MT regression model with the features of the RTE regression model to obtain a hybrid model (RTE+MT). We estimate the weights of all models from annotated translation corpora.

	BLEU	TER	MT	RTE	RTE+MT
NIST	60.0	64.0	65.6	63.1	68.3
WMT	35.9	37.5	39.6	42.3	45.7

Table 5.19: Correlation between human judgments and predictions on two corpora (Spearman’s ρ)

5.6.7.3. Experiment 1

We first test the metric on two publicly available corpora with human judgments. The first one, NIST, includes data from the Open MT Evaluations of 2006 and 2008. The second one, WMT, consists of the 2006 and 2007 datasets of the ACL/NAACL WMT workshops. We used ten-fold cross-validation for (regularized) parameter estimation and testing. Table 5.19 shows Spearman’s ρ correlation coefficients between the models’ predictions and human judgments. Results for the two corpora differ in absolute values, due, e.g., to the use of a 5-point adequacy scale (WMT) vs. a 7-point scale (NIST). However, the main tendencies are the same: MT and RTE outperform individual metrics, but are not clearly ordered. The hybrid system (RTE+MT) substantially outperforms all other systems. Apparently, MT evaluation benefits from the complementary types of information it obtains from shallow and entailment features.

5.6.7.4. Experiment 2

We submitted the RTE+MT model from Experiment 1, with feature weights estimated on the union of the NIST and WMT corpora, to the NIST MetricsMATR'08 single-reference track. The evaluation dataset was news and web data drawn from the NIST MT Eval and GALE programs, with some Transtac dialogue data. Across all performance analyses, our system was generally among the upper half. In the prediction of 7-point adequacy scores, RTE+MT obtained rank 14th of 39, with a correlation of $\rho = 0.61$ (best system: 0.68). In the prediction of qualitative scores, it attained 7th place ($\rho = 0.51$, vs. 0.57 for the best system). It was however the best system for the prediction of HTER scores ($\rho = -0.56$), the main GALE evaluation target. We hypothesize that the inferior performance in MetricsMATR compared to Exp. 1 is due to the following factors: (a), overfitting on training data due to a large number of parameters; (b), difficulty of structural features to deal with the often ungrammatical speech transcriptions in the Metric-sMATR data; (c), the absence of score normalization at the document or dataset level.

5.6.7.5. Conclusion

Our results show that the use of textual entailment features for MT evaluation is a promising perspective. We found that a vanilla RTE system — originally developed for well-formed English text — was mostly robust enough to compete with state-of-the-art MT evaluation metrics without adaptation. In particular, we found entailment-based features can be combined with traditional MT evaluation metrics to obtain a hybrid system that consistently outperforms either individual approach, which allows for the incremental improvement of evaluation schemes.

Our current metric still has a number of limitations which are the subject of on-going research. Currently, a large hurdle to its practical use is its high resource requirements. Our metric requires several seconds to score one segment pair. While this is fast enough to score large corpora for evaluation purposes, it is clearly too slow for integration into minimum error rate training (MERT). However, a number of simple changes can reduce runtime considerably. In addition, the independence between individual segment pairs makes scoring easily parallelizable.

References

- Agarwal, A. and A. Lavie. 2008. Meteor, M-BLEU and M-TER: Evaluation metrics for high-correlation with human rankings of machine translation output. *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 115–118, Columbus, Ohio, June. Association for Computational Linguistics.
- ALPAC (Automatic Language Processing Advisory Committee). 1966. Report of the ALPAC; Language and Machines: Computers in Translation and Linguistics. Division of Behavioral Sciences, National Academy of Sciences, National Research Council, Washington, D.C.
- Banerjee, S. and A. Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments. *Proceedings of the ACL 2005 Workshop on Intrinsic and Extrinsic Evaluation Measures for MT and/or Summarization*, pages 65–72, Ann Arbor, Michigan, June.
- Bannard, C. and C. Callison-Burch. 2005. Paraphrasing with Bilingual Parallel Corpora. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL 2005)*, pages 597–604, Ann Arbor, Michigan, June.
- Belvin, R.S., S. Riehemann and K. Precoda. 2004. A Fine-Grained Evaluation Method for Speech-to-Speech Machine Translation Using Concept Annotations. *Proceedings of the Fourth International Conference On Language Resources and Evaluation (LREC 2004)*, Lisbon, Portugal.
- Blatz, J., E. Fitzgerald, G. Foster, S. Gandrabur, C. Goutte, A. Kulesza, A. Sanchis and N. Ueffing. 2003. Confidence Estimation for Machine Translation. *Technical Report Natural Language Engineering Workshop Final Report*, Johns Hopkins University.
- Cahill, A., M. Burke, R. O'Donovan, J. van Genabith and A. Way. 2004. Long-distance dependency resolution in automatically acquired wide-coverage PCFG-based LFG approximations. *Proc. ACL*, pages 319–326.
- Callison-Burch, C., M. Osborne and P. Koehn. 2006. Re-Evaluating the Role of Bleu in Machine Translation Research. *Proceedings of EACL-2006*, pages 249–256, Trento, Italy.
- Callison-Burch, C., C. Fordyce, P. Koehn, C. Monz and J. Schroeder. 2007. (meta-) evaluation of machine translation. *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 136–158, Prague, Czech Republic, June. Association for Computational Linguistics.
- Charniak, E. and M. Johnson. 2005. Coarse-to-fine n-best parsing and maxent discriminative reranking. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 173–180, Ann Arbor, Michigan, June. Association for Computational Linguistics.
- Child, J.R., T. Ray and P. Lowe, Jr. 1993. Proficiency and Performance in Language Testing. *Applied Language Learning*, Vol. 4.

- Church, K. and E. Hovy. 1993. Good applications for crummy machine translation. *Machine Translation*, 8:239–258.
- Coughlin, D. 2003. Correlating Automated and Human Assessments of Machine Translation Quality. *Proceedings of MT Summit IX*, pages 63–70, New Orleans, LA.
- Dagan, I., O. Glickman and B. Magnini. 2006. The PASCAL Recognising Textual Entailment Challenge. Quiñero-Candela, J., Dagan, I., Magnini, B., d'Alché-Buc, F. (Eds.) *Machine Learning Challenges. Lecture Notes in Computer Science*, Vol. 3944, pp. 177-190, Springer.
- Doddington, G. 2002. Automatic Evaluation of Machine Translation Quality Using N-gram Co-Occurrence Statistics. *Proceedings of the Human Language Technology (Notebook)*, pages 128–132, San Diego, CA.
- Fellbaum, C. 1998. WordNet: An Electronic Lexical Database. MIT Press. <http://www.cogsci.princeton.edu/~wn> [2000, September 7].
- Fisher, F. and C.R. Voss. 1997. Falcon, an mt system support tool for nonlinguists. *Proceedings of the Advanced Information Processing and Analysis Conference (AIPA 97)*, pages 182–191, McLean, VA.
- Fisher, F., C. Schlesiger, L. Decrozant, R. Zuba, M. Holland and C.R. Voss. 1999. Searching and translating arabic documents on a mobile platform. *Proceedings of the Advanced Information Processing and Analysis Conference (AIPA 99)*, Washington, DC.
- Frederking, R. and S. Nirenburg. 1994. Three Heads are Better than One. *Proceedings of the Fourth Conference on Applied Natural Language Processing (ANLP-94)*, Stuttgart, Germany.
- Garofolo, J.S., T. Robinson, and J. G. Fiscus. 1994. The development of file formats for very large speech corpora: Sphere and shorten. *Proceeding of ICASSP*.
- Gates, D., A. Lavie, L. Levin, A. Waibel, M. Gavalda, L. Mayfield, M. Woszczyna and P. Zhan. 1996. End-to-End Evaluation in JANUS: a Speech-to-Speech Translation System. *Proceedings of the European Conference on Artificial Intelligence (ECAI-1996)* (Workshop on “Dialogue Processing in Spoken Language”), Budapest, Hungary, August.
- Gimenez, J. and L. Marquez. 2007. Linguistic features for automatic evaluation of heterogeneous MT systems. *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 256–264, Prague, Czech Republic, June. Association for Computational Linguistics.
- Habash, N. and A. Elkholy. 2008. Sepia: Surface span extension to syntactic dependency precision-based mt evaluation. *Proceedings of the NIST Metrics for Machine Translation Workshop at the Association for Machine Translation in the Americas conference, AMTA-2008*, Waikiki, Hawaii.
- Hutchins, W.J. 2001. Machine Translation Over Fifty Years. *Histoire Épistémologie Langage*, I(23):7–31. 2008. ILR language skill level descriptions. <http://www.govtilr.org>.

- Jones, D. and W. Shen. 2006. Two New Experiments for ILR-Based MT Evaluation. *Proceedings of Association for Machine Translation in the Americas*.
- Jones, E., T. Oliphant, P. Peterson, *et al.* 2001–. *SciPy*: Open source scientific tools for Python.
- Jones, D.A., W. Shen, N. Granoien, M. Herzog and C. Weinstein. 2005. Measuring translation quality by testing English speakers with a new defense language proficiency test for Arabic. *Proceedings of 2005 International Conference on Intelligence Analysis*, May 2-6, 2005, McLean, VA., May.
- Jones, D., M. Herzog, H. Ibrahim, A. Jairam, W. Shen, E. Gibson and M. Emonts. 2007. ILR-based MT comprehension test with multi-level questions. *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 77–80, Rochester, New York, April. Association for Computational Linguistics.
- Joshi, A.K. and Y. Schabes. 1997. Tree-adjointing grammars. *G. Rozenberg and A. Salomaa, eds., Handbook of Formal Languages*, Volume 3: Beyond Words, pages 69–124. Springer, New York.
- Kauchak, D. and R. Barzilay. 2006. Paraphrasing for Automatic Evaluation. *Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL*, pages 455–462.
- King, M. 1996. Evaluating Natural Language Processing Systems. *Communication of the ACM*, 29(1):73–79, January.
- Knight, K. and I. Chander. 1994. Automated Postediting of Documents. *Proceedings of National Conference on Artificial Intelligence (AAAI)*, pages 779–784, Seattle, Washington.
- Laoudi, J., C. Tate and C.R.Voss. 2006. Task-based mt evaluation: From who/when/where extraction to event understanding. *Proceedings of the Fifth International Conference on Language Resources and Evaluation*, LREC-2006, pages 2048–2053, Genoa, Italy.
- Lavie, A. and A. Agarwal. 2007. METEOR: An automatic metric for MT evaluation with high levels of correlation with human judgments. *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 228–231, Prague, Czech Republic, June. Association for Computational Linguistics.
- Lavie, A., K. Sagae and S. Jayaraman. 2004. The Significance of Recall in Automatic Metrics for MT Evaluation. *Proceedings of the 6th Conference of the Association for Machine Translation in the Americas (AMTA-2004)*, pages 134–143, Washington, DC, September.
- Leusch, G., N. Ueffing and H. Ney. 2006. CDER: Efficient MT Evaluation Using Block Movements. *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2006)*.

- Levenshtein, V. I. 1966. Binary Codes Capable of Correcting Deletions, Insertions, and Reversals. *Soviet Physics Doklady*, 10:707–710.
- Levin, L., D. Gates, A. Lavie, F. Pianesi, D. Wallace, T. Watanabe and M. Woszczyna. 2000. Evaluation of a Practical Interlingua for Task-Oriented Dialogue. *Proceedings of ANLP/NAACL-2000 Workshop on Applied Interlinguas*, pages 18–23, Seattle, WA.
- Liu D. and D. Gildea. 2005. Syntactic features for evaluation of machine translation. *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 25–32, Ann Arbor, Michigan.
- Lopresti, D. and A. Tomkins. 1997. Block edit models for approximate string matching. *Theoretical Computer Science*, 181(1):159–179, July.
- Madnani, N., N.F. Ayan, P. Resnik and B.J. Dorr. 2007. Using paraphrases for parameter tuning in statistical machine translation. *Proceedings of the Workshop on Statistical Machine Translation*, Prague, Czech Republic, June. Association for Computational Linguistics.
- Madnani, N., P. Resnik, B.J. Dorr and R. Schwartz. 2008. Are Multiple Reference Translations Necessary? Investigating the Value of Paraphrased Reference Translations in Parameter Optimization. *Proceedings of the Eighth Conference of the Association for Machine Translation in the Americas*, October.
- Magerman, D.M. 1995. Statistical decision-tree models for parsing. *Proc. ACL*, pages 276–283.
- Matusov, E., G. Leusch, O. Bender and H. Ney. 2005. Evaluating machine translation output with automatic sentence segmentation. *Proceedings of the International Workshop on Spoken Language Translation (IWSLT) 2005*.
- Moore, R. C. and C. Quirk. 2008. Random restarts in minimum error rate training for statistical machine translation. *Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008)*, pages 585–592, August.
- Nelder, J. A. and R. Mead. 1965. A simplex method for function minimization. *Computer Journal*, 7:308–313.
- Nießen, S., F.J. Och, G. Leusch, and H. Ney. 2000. An evaluation tool for machine translation: Fast evaluation for MT research. In *Proceedings of the 2nd International Conference on Language Resources and Evaluation (LREC-2000)*, pages 39–45.
- NIST and LDC. 2007. Post Editing Guidelines for GALE Machine Translation Evaluation, Version 3.0.2, May 25.
- Nübel, Rita. 1997. End-to-End Evaluation in VERBMOBIL I. *Proceedings of MT Summit VI*, pages 232–239, San Diego, CA.
- Och, F.J. and H. Ney. 2002. Discriminative training and maximum entropy models for statistical machine translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 295–302, Philadelphia, PA, July.

- Och, F.J.. 2003. Minimum Error Rate Training in Statistical Machine Translation. *Proceedings of the 41st Annual Meeting of the Association of Computational Linguistics*, pages 160–167, July.
- Owczarzak, K., J. van Genabith and A. Way. 2007. Labeled dependencies in machine translation evaluation. *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 104–111, Prague, Czech Republic. Association for Computational Linguistics.
- Papineni, K., S. Roukos, T. Ward and W. Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 311–318, Philadelphia, PA.
- Porter, M.F. 1980. An algorithm for suffic stripping. *Program*, 14(3):130–137.
- Powell, M. J. D. 1964. An efficient method for finding the minimum of a function of several variables without calculating derivatives. *Computer Journal*, 7:155–162.
- Przybocki, M., K. Peterson and S. Bronsart. 2008. Official results of the NIST 2008. “Metrics for MACHINE TRANSLATION” Challenge (Metrics-MATR08). <http://nist.gov/speech/tests/metricstr/2008/results/>, October.
- Roark, B., M. Harper, E. Charniak, B. Dorr, M. Johnson, J.G. Kahn, Y. Liu, M. Ostendorf, J. Hale, A. Krasnyanskaya, M. Lease, I. Shafran, M. Snover, R. Stewart and L. Yung. 2006. SParseval: Evaluation metrics for parsing speech. *Proc. LREC*.
- Rosti, A.V., S. Matsoukas and R. Schwartz. 2007. Improved word-level system combination for machine translation. *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 312–319, Prague, Czech Republic, June. Association for Computational Linguistics.
- Russo-Lassner, G., J. Lin and P. Resnik. 2005. A Paraphrase-Based Approach to Machine Translation Evaluation. *Technical Report LAMPTR-125/CS-TR-4754/UMIACS-TR-2005-57*, University of Maryland, College Park.
- Sanders, G.A., S. Bronsart, S. Condon and C. Schlenoff. 2008. Odds of Successful Transfer of Low-level Concepts: A Key Metric for Bidirectional Speech-to-speech Machine Translation in DARPA’s TRANSTAC Program. *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC-2008)*, Marrakesh, Morocco, May. European Language Resources Association (ELRA).
- Snover, M., B. Dorr and R. Schwartz. 2004. A Lexically-Driven Algorithm for Disfluency Detection. *Proceedings of HLT/NAACL*, pages 157–160.
- Snover, M., B. Dorr, R. Schwartz, L. Micciulla and J. Makhoul. 2006. A Study of Translation Edit Rate with Targeted Human Annotation. *Proceedings of Association for Machine Translation in the Americas (AMTA-2006)*, pages 223–231, Cambridge, Massachusetts.

- Tate, C.R. and C.R. Voss. 2006. Combining evaluation metrics via loss functions. *Proceedings of the Association for Machine Translation in the Americas conference*, AMTA-2006, Boston, MA.
- Tate, C.R. 2007. An Investigation of the Relationship between Automated Machine Translation Evaluation Metrics and User Performance on an Information Extraction Task. *Ph.D. thesis, University of Maryland*, College Park, MD.
- Tate, C.R. 2008. A statistical analysis of automated mt evaluation metrics for assessments in task-based mt evaluation. *Proceedings of the Association for Machine Translation in the Americas conference*, AMTA-2008, pages 182–191, Waikiki, Hawaii.
- Taylor, K. and J. White. 1998. Predicting what mt is good for: User judgments and task performance. *Proceedings of the Association for Machine Translation in the Americas conference*, AMTA-1998, Langhorne, PA.
- Tillmann, C., S. Vogel, H. Ney, A. Zubiag and H. Sawaf. 1997. Accelerated DP Based Search For Statistical Translation. *European Conference on Speech Communication and Technology*, pages 2667–2670, Rhodes, Greece, September.
- Turian, J.P., L. Shen and D.I. Melamed. 2003. Evaluation of machine translation and its evaluation. *Proc. MT Summit IX*, pages 386–393, New Orleans, LA.
- Vanni, M., C.R. Voss and C.R. Tate. 2004. Ground truth, reference truth and “omniscient truth” – parallel phrases in parallel texts for mt evaluation. *Proceedings of the Fourth International Conference On Language Resources and Evaluation (LREC 2004)*, pages 10–13, Lisbon, Portugal.
- Voss, C.R. and C.R. Tate. 2006. Task-Based Evaluation of Machine Translation (MT) Engines: Measuring How Well People Extract Who, When, Where Type Elements in MT Output. *Proceedings of the 11th Annual Conference of the European Association for Machine Translation (EAMT-2006)*, pages 203–212, Oslo, Norway.
- White, J.S. and T. O’Connell. 1994. Evaluation in the ARPA Machine Translation Program: 1993 Methodology. *Proceedings of the ARPA HLT Workshop*, Plainsboro, NJ.
- White, J.S., T. O’Connell and F. O’Mara. 1994. The ARPA MT Evaluation Methodologies: Evolution, Lessons, and Future Approaches. *Proceedings of the First Conference of the Association for Machine Translation*, pages 193–205.
- White, J.S., J.B. Doyon and S.W. Talbott. 2000. Task Tolerance of MT Output in Integrated Text Processes. *ANLP/NAACL 2000: Embedded Machine Translation Systems*, pages 9–16.
- Wilks, Y. 2008. *Machine Translation: Its Scope and Limits*. Springer Verlag, New York, NY.
- Zhou, L., C.Y. Lin and E. Hovy. 2006. Re-evaluating Machine Translation Results with Paraphrase Support. *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, pages 77–84.

Chapter 6: Operational Engines

Editors: Douglas W. Oard, Carl Madson, Joseph Olive, John McCary, and Caitlin Christianson

6.1 Introduction

The goal of the GALE Program is to empower the warfighter through the use of language technologies. In order to accomplish this goal, the technologies must be integrated into operational engines that meet the needs of both warfighters and those who support them. That integration imperative leads directly to the two key parts of this chapter: how to perform that integration, and assessing how well we have met operational needs.

The chapter begins with a look at the deployment history over the past decade and a half for speech and language technologies in operational environments. As Kathleen Egan and Allen Sears (Section 6.2.1) explain, this has been an evolutionary process driven by an increasingly rich understanding of warfighter needs. The second contribution provides details on the design of an operational engine that has recently been deployed in operational environments around the world, including in direct support of operational forces in the field. As Daniel Kiecza and his colleagues (Section 6.2.2) explain, meeting the need for responsive and accurate performance by integrating systems that each have a different development heritage can be a substantial undertaking, sometimes requiring development of new technologies. As an example of that process, Amit Srivastava and Daniel Kiecza (Section 6.2.3) describe the development of a new technique for establishing sentence boundaries during speech recognition that substantially improves the latency for subsequent translation of the recognized speech.

The integration challenge in GALE extends well beyond pairing components, however. John Pitrelli and his colleagues (Section 6.2.4) introduce the full scope of the challenge by describing the GALE Interoperability Demonstration (IOD) system. Coordinating real-time processing by eleven different types of language technologies would be a substantial undertaking under any circumstances; doing so in a distributed manner using the facilities of seven research groups spread over thousands of miles in three countries raises the challenge to an entirely new level. But, it is exactly that level at which we must learn to work if we are to learn how best to integrate the diverse language technologies that we are now able to create.

The core technology behind the GALE IOD system is the Unstructured Information Management Architecture (UIMA), a flexible integration platform designed to meet the unique demands of distributed language processing. Eric Nyberg (Section 6.2.5) and his colleagues describe how UIMA-enabled components can be shared using GALE's UIMA Component Repository (UCR), and how individual researchers can flexibly configure entire UIMA processing environments using GALE's UIMA Component Container (UCC).

The unprecedented complexity of the systems that we can now assemble poses substantial challenges for evaluation. We know from intrinsic evaluation how well individual components, and even specific types of component combinations, meet our expectations for their behavior. However, calibrating our expectations for parts of a complex system against the true needs of users of the entire system is another matter altogether. No one approach can provide a comprehensive answer to every question we must answer about the utility of the systems that we build, so in the second part of this chapter we look at three ways that GALE researchers have gained such insights. Daqing He and his colleagues begin that process by describing a series of user studies to evaluate the utility of specific ways of integrating language technologies by using the Evaluation Dataset for Integration Exploration (EDIE) that they developed for GALE to create a controlled evaluation environment. Studies of this type by each GALE team offered a principled basis for system design decisions.

To bring GALE systems from the research lab to the warfighter requires that we augment these initial controlled user studies with additional user studies in representative environments and with representative users. Douglas Oard and his colleagues tackle the challenge of evaluating a fully integrated system in a representative environment by running a rich set of formative evaluation studies using live content, designing tasks based on real-world events as they were unfolding. Connie Fournelle and her colleagues then put experienced intelligence analysts in front of fully integrated GALE systems and challenged them with realistic tasks. Never before has it been possible to deliver this level of integrated language processing to real analysts, and the results of this important study have served to ground resource allocation decisions throughout the remainder of the GALE program.

As with many difficult challenges, robust investment in system integration is, perhaps, more often preached than practiced. We have forces deployed in harm's way in the service of our nation, however, and those brave men and women demand and deserve our best effort to create the robust and effective integrated systems that they accomplish what we have asked of them. Unprecedented challenges call for innovative approaches, and our goal in this chapter is to describe how the GALE research teams have risen to meet that challenge.

6.2 Implementation of Operational Engines

6.2.1. Bridging the Gap between Research Advances and Operational Users

Authors: Kathleen Egan and Allen Sears

6.2.1.1. Introduction

The efforts in bringing new technologies to bear in an operational setting started with the first meaningful Arabic data collection in 1998, including a variety of Arabic media sources, such as Al Jazeera. Established in 1996 with a grant from the Emir of Qatar, Al Jazeera's BBC-trained staff brought the styles and techniques of Western media to the

Arabic-speaking world for the first time. Covering the politics and conflicts of the region with a relatively apolitical eye, Al Jazeera accomplished the feat of appearing to be both anti-American (from the point of view of the U.S.) and anti-establishment (from the point of view of regional governments and powers). It was Al Jazeera's airing of Al Qaeda's home-produced video statements that brought the station worldwide attention.

Because there were so many media outlets to monitor (including a rapidly increasing number of Web pages and blogs), by 2000 it became clear that U.S. personnel could no longer keep up with the inflow of media at all levels. No automation existed to stem the flow, although various organizations attempted to handle it by force of human effort. In addition, the vast majority of the content was in languages outside the Cold-War stalwarts of Russian, German, and the like, more than doubling the challenge to keeping up using human translators.

The Department of Defense (DoD) graduates approximately 2500 linguists each year, of whom about 750 specialize in Middle Eastern languages. Today these numbers satisfy only 10-15% of the military's requirements for interpreters and translators. The current requirement is for about 10,000 linguists. With a rough annual cost of training each individual at approximately \$250,000, the military cannot afford to train enough qualified linguists to support today's needs (Gilmore 2007). Moreover, even if enough linguists in all the priority languages were available, the amount of data to be analyzed is growing exponentially, and humans alone cannot monitor all of the available data in a timely fashion.

Language is at the heart of human communication and having timely and accurate translations can save lives. Words are powerful. They have infuriated populations, created misunderstandings, fueled hatred, and built barriers across cultures and civilizations. At the same time, words have opened doors, reconciled groups, gained peace, and created laws for justice, order, democracy and freedom. The imperative with the sea of words from around the world is for accuracy, but more importantly, the meaning of each word in context.

In the past ten years, human language technology (HLT) insertion in user-friendly solutions stepped in to bridge the two major capabilities gaps afflicting U.S. operations: **media triage** and **foreign language comprehension**. Various government agencies are now leveraging GALE advances and building on the on-going work of the Technical Supporting Working Group (TSWG). TSWG is an interagency research and development arm of the Office of the Secretary of Defense that is funding media monitoring development to enable the warfighter to gain immediate actionable intelligence from open source media.

6.2.1.2. Pre-GALE Efforts: Early Applications of HLT in the Intelligence Community (IC) and DoD

Throughout the 1990s, DARPA funded basic research in speech recognition (sometimes known as speech-to-text) and machine translation. While incremental progress had been made in the accuracy of transcription and translation during that decade, transition of these technologies to operational use had occurred only on a piecemeal basis. Users of these technologies were largely responsible for envisioning a concept of operations that would be introduced into their existing business processes, all

of which assumed manual, human-intensive approaches. As such, technology transition was limited, and early adopters were not fully supported in the insertion of advanced language processing technologies into operational settings. Most user groups did not have strong information technology support, and did not want to depend on unreliable machines in support of operational tasks. Progress in transition has improved over the years through the adoption of a strategy of testing technologies with proxy users and then engaging them to define and improve on the proposed concept of operations. However, it is only when moving technology from an experiment to an operational prototype and later to a fully integrated system in the workplace that full transition and technology maturity occurs.

Before 2000, the overall technical goal for automated language processing was to reduce word error rates for speech, and achieve fluency and accuracy measures for machine translation quality. A shift started around 2000, with a goal of supporting users in the language processing tasks with which they were faced on a daily basis. From 2000 to 2003, three major efforts in the IC and DoD laid the groundwork for bridging the gap between a community of users who relied heavily on human capital for all of its analysis, and a research community that focused on the hard scientific problems of speech recognition and machine translation.

OASIS, TAP projects, and MAPS were the predecessors to the broadcast monitoring and Web monitoring capabilities that now support language needs and are currently deployed in a growing number of operational sites.

6.2.1.3. OASIS

In 2000, the Intelligence Community wished to broaden and accelerate the coverage of open-source broadcast media to Government decision makers. The Foreign Broadcast Information Service (FBIS, now known as the Open Source Center) was responsible for media processing that was extremely labor-intensive. The sources were non-English, and native speakers performed selection, transcription, and translation of the audio sources. Selection was the process of locating media that was “newsworthy” for dissemination. Transcription was the process of transforming the media into typewritten form in the source language, and translation converted the non-English transcript into English for all selected excerpts.

FBIS was interested in automating the processing pipeline, starting with the selection process, and considered using automatic speech recognition as a solution that would turn a listening problem into a reading or skimming problem, and eventually into a searching problem. Furthermore, having a transcript as a starting point would speed up the eventual translation of the media. To test this theory, BBN was awarded a contract to develop the OASIS system and use technology that BBN had been developing for DARPA called “Rough and Ready” (Kubala *et al.* 2000). OASIS allowed the scheduled recording and transcription of broadcast-news-quality audio, initially in South Asian-accented English, and later also in Arabic and Mandarin Chinese. OASIS was deployed to several international customer locations for use by local media analysis staff.

OASIS proved to be highly successful at this task. While initially regarded as a threat by the local translators, the system acted as an additional set of “digital ears”, monitoring stations that the customer wished to cover, but for which they lacked staff to conduct

real-time reviews. OASIS recorded and transcribed stations on a time-shift basis, then allowed rapid review of the captured content for intelligence value. In a 2000 Weekly Activity Report (WAR) to the Secretary of Defense, DARPA reported that automatic language processing technology had been licensed by the commercial sector and would be used for automatic indexing of audio information, such as radio broadcasts. Basic speech recognition and information extraction components came from previous DARPA investment in the Human Language Systems program.

The users concluded that being able to view the text while gisting the most important excerpts of the news made it possible for analysts to cover a larger number of hours of captured media in substantially less time. During these early OASIS prototypes testing, analysts reported that the word error rate did not bother them as long as they could corroborate the accuracy of the transcription by listening to the audio. This was important feedback from users that helped shape technology development priorities and user interface.

OASIS marked the first major successful use of human language technology to bridge the media triage gap in an operational context for an operational government customer (Shepard *et al.* 2002).

6.2.1.4. Text and Audio Processing (TAP) Experiments

In late 2003, the DARPA TIDES program was focused on using HLT to enable native English speakers to access and process non-English media content with the same ease as English media. The TIDES program held regular pseudo-operational evaluations known as Integrated Feasibility Experiments, or IFEs, allowing proxy users to work with the technology developed by the research community in a controlled environment. Three organizations participated in the TAP experiments and helped shape the technology from their own perspectives. MITRE developed the MiTAP system that focused on information management of language reports on the Web regarding identifying and tracking disease outbreaks. Both Virage and BBN developed experimental systems (ViTAP and OnTAP respectively) to harvest and translate open-source Web content from news sites. ViTAP and OnTAP allowed analysts to write reports based on the content processed by the system. Both systems were given operational prototype stress-tests in 2004 at the Strong Angel II humanitarian response technology demonstration.

In September 2001, the world's attention shifted to the 9/11 attacks by terrorists on the U.S. The DARPA HLT programs, which had been developing English transcription and translation capability, immediately focused all their effort on Arabic. Terrific progress was made on automated Arabic language processing in 2002 and 2003, and by the end of 2003, DARPA was ready to conduct an operational prototype experiment with an Arabic system. In December 2003, DARPA management approved an "experimental Text Information Retrieval (eTIR)" project that was based on the emerging TAP technologies.

DARPA, with support from the National Virtual Translation Center (NVTC), developed an eTAP initiative with the J2 Open Source Cell at United States Central Command (CENTCOM) in Tampa, Florida. The Open Source (OSINT) Cell was charged with collecting and reporting on Middle Eastern media coverage of U.S. actions in Iraq and Afghanistan for the CENTCOM commander. Their product was known as the

“Foreign Media Perceptions Digest” and was published three times a week. The technology at their disposal consisted of a set of America Online accounts, which they used to review English Web sites reporting on news in the CENTCOM theater. The OnTAP technology was selected to be modified and applied to the OSINT task for the eTIR project.

With support from DARPA, BBN developed an operational prototype variant of OnTAP called eTAP, for CENTCOM’s exclusive use. eTAP collected Web pages from Arabic language news Web sites, translated them using a machine translation engine from Language Weaver, Inc., and made them searchable in English through a simple user interface. The system was built to provide a platform on which to develop a CONOPS for automated language processing in support of the Enduring Freedom effort in Iraq, and to measure the contribution of the technology in response to the operational needs. Even though the machine translation capability was far from perfect, the hypothesis was that the current state of the art in machine translation was of sufficient quality to allow accurate gisting of open-source content.

The eTAP system allowed the CENTCOM users to set up personalized search terms, called a “watchlist,” and review them at will. Using the rough machine translation to judge the relevance of an article, the users could click an on-screen button, which would task a human linguist at the NVTC in Washington, D.C., to create a more accurate, human-vetted translation. This was performed first overnight, then in less than two hours if deemed critical, and returned to the analysts at CENTCOM for use in their reports.

The eTAP capability proved to be extremely successful. During the first six months of operational use, the system harvested over 1.5 million web pages. CENTCOM staff selected over 1,300 automated translations for human gisting, and used over 99% of them in their final products, rejecting only two translations because they were either not relevant or redundant. As a direct result of their new level of success, CENTCOM increased its publication rate from three news roundup products per week to multiple daily products and expanded its staff to more than ten analysts and linguists. Growth in demand occurred in part due to the increased visibility brought about by the technological assist; instead of reducing overall labor, the system helped to create a capability. The use of eTAP at CENTCOM marked the first automated system that provided foreign language comprehension to the active warfighter.

6.2.1.5. MAPS

In 2003, TSWG responded to a requirement from the Deputy Directorate for Information Operations (DDIO), a component of the Joint Chiefs of Staff J2/J3 Operations Directorate based in the Pentagon. The DDIO was focused on winning the international information campaign against terrorism. Their primary function was to monitor, collect, and resolve conflicting information from domestic and international broadcasts that was relevant to U.S. military interests. Timeliness of the collected information was of the essence.

The DDIO had little automation support; staff members kept televisions on all the time, waiting for relevant broadcasts to air. When an important story appeared on the screen, staff members and linguists would run into a media room equipped with a set of digital video recorders to attempt to translate the news program. The DDIO was seeking

automated methods to allow them to monitor an increasing amount of media with the same or fewer staff.

IC and DoD entities have always monitored foreign media using language analysts and/or native speakers to watch the news and report on critical events. This human approach was time and resource intensive, leading to a strong requirement to add support and go beyond the human effort. TSWG responded to DDIO need by capitalizing on the research gains realized by DARPA and moving them into an operational context, with real users who would rely on the technology for their day-to-day work. The practice of humans watching television news and reporting the highlights at the end of the day was deemed insufficient and unpractical when timely responsiveness and documented evidence of facts were paramount in gaining situational awareness and combating terrorism. The realization that human-machine collaboration was a better approach for timely handling of the volume of data motivated the subsequent development, architecture, configuration, integration and deployment of systems in operational environments.

TSWG leveraged OASIS and funded a team led by BBN to develop and deploy the Multimedia Alert Processing System (MAPS), delivered to the DDIO as a prototype in two phases. The initial prototype supported monitoring of two English and three Arabic television channels. The second delivery incorporated upgrades and features requested by the user community. MAPS was the first turnkey application that was physically deployed in two operational settings (DDIO from 2003 to 2004 and CENTCOM from 2003 to the present) combining audio, text, video, and translingual search into one application.

TSWG funded extensions and improvements in response to specific requirements from users at the CENTCOM J2 OSINT section targeted at increasing its analysts' ability to access Arabic-language media. These requirements included expanding its MAPS system to two channels and enhancing the eTAP Web harvesting system.

At the same time, three overseas deployments of MAPS took place in support of Operation Iraqi Freedom: one with the Department of State Multi National Forces - Iraq (MNF-I), one with PsyOps, and one with Special Forces.

6.2.1.6. TALES (Translingual Advanced Language Exploitation System)

In 2005, the Joint Intelligence Center PACIFIC (JICPAC) and DARPA planned a field test experiment to exploit web and broadcast information coming from the Pacific area of operations. DARPA selected and funded IBM for the experiment and decided that the focus should be on Chinese transcription and translation. IBM was to develop and insert a prototype system to be called TALES. The hardware and software were to be packaged in a "deployment unit" installed at JICPAC in Hawaii. Open source processing personnel at JICPAC were given the task of developing analytical models to predict the possible impact of automated language exploitation capability. In addition, the DARPA/GALE Language Exploitation Environment (LEE) effort would demonstrate capabilities through the use of IBM's UIMA (Unstructured Information Management Architecture) framework. The objective was to test data management and technical integration point approaches in order to discover both strengths and weaknesses relevant to future JICPAC operations.

The plan was for DARPA and JICPAC to develop a strategic relationship based on leveraging current language processing components within the UIMA framework to offer PACCOM an open-source processing capability that could be expanded to support requirements for an increasing volume of Chinese translation while reducing turn-around time and improving open source products. However, one lesson learned from the TALES insertion effort was that there was a delicate balance in opportune timing between capability maturity and technology insertion, as highlighted by the relative lack of maturity and capability of Chinese audio and text processing. At the time, the research capabilities for Chinese language processing were not sufficient to be of assistance to operational users. DARPA learned about the shortcomings in Chinese processing techniques from TALES and from metrics-based testing. This led to increased efforts to find new approaches aimed at improving automated translation. Chinese translation techniques are still a high priority for DARPA/GALE today.

6.2.1.7. BMS and WMS

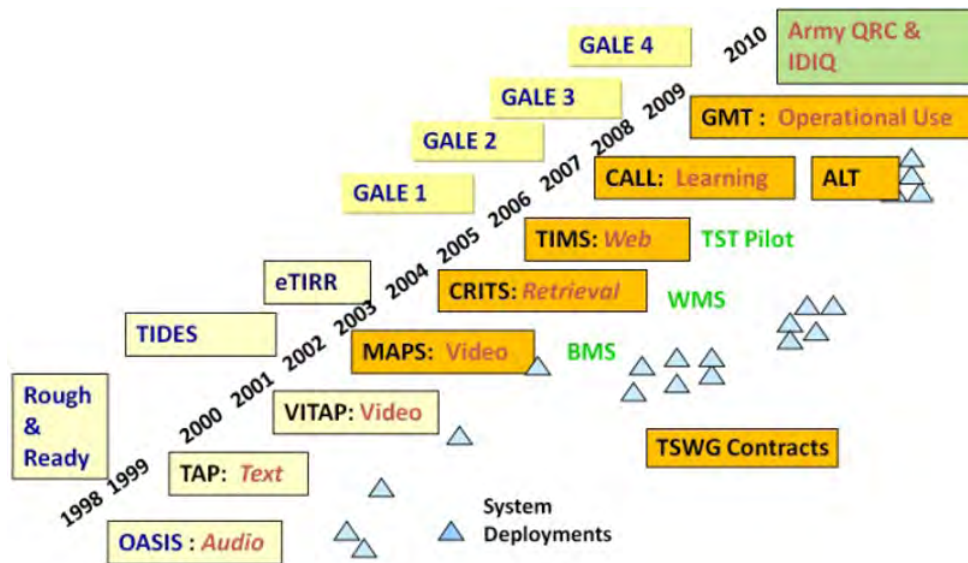


Figure 6.1: Timeline of HLT R&D efforts

The approach adopted in these efforts was to mitigate machine errors and focus on robust retrieval methods, maximizing the users' expertise by providing intuitive, user-friendly interfaces and productivity tools that supported the users' day-to-day operations. TSWG insisted that development be closely linked with efforts to observe, listen to, and respond as a team to the unique operational needs of users. A key objective was to exploit open-source media in order to efficiently and effectively monitor the growing impact of that media on military and intelligence analysis and actions. Both the BMS and WMS

helped to moderate the compounding challenges of access to the growing volume of data and the limited number of analysts and hours in the day to complete operational tasks.

Since 2005, the BMS and WMS have been available as fully-functional, turnkey systems that can be deployed to customer sites in a leave-behind mode for operational use. Since these two systems have provided the most successful insertions for GALE technology to date, a short review of the applications follows:

The BBN Broadcast Monitoring System creates a continuous, searchable archive of foreign language television broadcasts. For a complete description of BMS, see Section 6.2.2.6.

The BMS allows users with no foreign language skills to get the gist of a broadcast and triage enormous volumes of media, allowing linguists to focus on translation tasks. Users can quickly find spoken content in the video archive through keyword queries — in English or the source language — and play back video from any point in the system’s one-year internal cache. Searches can be saved in a watch-list and are automatically populated by the system as new matches are found.



Figure 6.2: WMS article view

The BBN Web Monitoring System (WMS) is an end-to-end system providing the capability of collecting, organizing, and translating open-source content from the World Wide Web. The WMS integrates and manages the workflow of the media analysis process from beginning to end — from data collection and processing, to automated triage and retrieval, to machine-assisted translation and support for human translation, to publication and dissemination.

WMS continuously captures content from user-selected Web sites and stores it in an archive that can be shared by multiple, distributed user groups. The captured site is archived and versioned for later use.

WMS supports over 30 input languages. Pages are automatically translated into English using machine translation software from Language Weaver. English speakers can use the machine translation to get the gist of an article, and then reach through the network for high quality human translation support.

In addition to the insertion of GALE advances in speech and machine translation through system upgrades, we have found that developing assistive tools to improve workflow is critical. Users differ in their needs for the output of the data. By working with them directly, we have been able to adjust the applications to fit the users' tasks and facilitate user acceptance of the technology.

6.2.1.8. User Feedback

Technology transition to a community of operational users involves a delicate balancing act. The natural tendency for technology providers is to attempt to field the most aggressive, highest performance application that represents the fruit of the labor of the research community. Yet, what leads to rapid adoption tends to be much different. TSWG insertion experiments have demonstrated that rapid adoption is best achieved if applications are simple to use and integrate seamlessly with users' tasks. Moreover, prototype systems must be robust and require little in the way of maintenance. The success of the TSWG/DARPA partnership lies in the ability to address both of these seemingly conflicting goals, resulting in applications that can be fielded to environments as harsh as a dusty machine room in Iraq, and still result in measurable improvements in military operations.

Over the past ten years, in successive deployment of automated language technologies, the key lessons learned have been that users care about quality and give the highest priority to applications that support their operations. The successes of the TSWG/DARPA partnership can be attributed to the rapid transition of research engines improved by GALE algorithms into useful applications. Human language technologies, such as speech-to-text and machine translation, by themselves are not sufficient to empower users to perform analytical tasks. Rather, these technologies are critical enablers of knowledge discovery, data exploitation, trend analysis, and product creation. Military and other Government users, including both language professionals and operators with no foreign language skills, must find an application to be user friendly and to match their operational workflow.

Starting in 2004, the CENTCOM Open Source Intelligence Cell became the true test bed for a full operational insertion of HLT. The experience deploying the WMS and BMS at CENTCOM represents a case study of this effect. Since 2004, the CENTCOM J2 Open Source Cell has grown from three users creating a thrice-weekly handcrafted product to a 24x7 operation with more than 25 analysts, creating multiple daily products and servicing in excess of 2,500 requests for information annually. Functioning as a stateside laboratory, both the WMS and BMS evolved based on requirements that were expressed by the CENTCOM user community and then rolled into the technology baseline.

Examples of products enabled by this technology include: responses to requests for information (RFIs), daily Foreign Media Trackers, weekly Pandemic Threat Updates, weekly Friday Sermon Analyses, weekly Threat Network Bulletins, and many more that are sensitive but critical to operations. As an indicator of success, CENTCOM started to anticipate RFIs, and started producing CIPs (Critical Information Products) to answer critical information needs before they had even been requested.

CENTCOM management has stated that:

Our whole operation is based around these [foreign language processing] technologies and without them we would have to rebuild our operation from scratch.”

Figure 6.3 shows the quantity of RFIs and CIPs satisfied by CENTCOM staff from 2005 through September 2009.

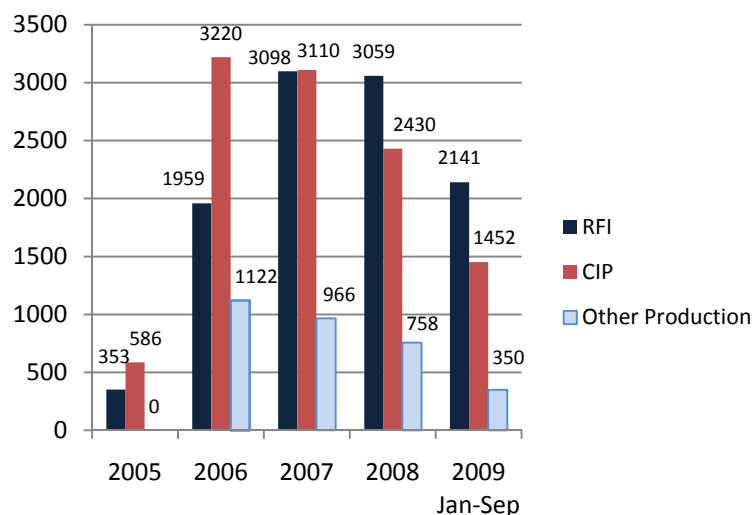


Figure 6.3: CENTCOM RFI and CIP Production, 2005-09

Working hand-in-hand with users in domestic command center environments, such as that found in Tampa at CENTCOM, helped shape capabilities and CONOPS, which in turn increased the odds of success in more hostile environments. For example, deployments to Iraq were successful as a result of the intense insertion experiments in Tampa. In fact, four deployments of BMS and WMS to units in Iraq occurred from 2005 through 2009. In all cases, deployments were handled by contractors with minimal pre-deployment contact with the fielding units. Hostile conditions are taxing and challenging. For example, reliable power is rare, with abrupt power failures regularly damaging components. In addition, audio/visual and internet capabilities are obtained by receiving units through local companies in neighboring towns in Iraq. Another serious problem is the fact that units rotate back to the United States every six to twelve months, with virtually no handover to the incoming unit.

Despite these challenges, user acceptance has been extremely high for the BMS and WMS. One site in Baghdad reports that:

BMS was a true labor saver. With dozens of Pan Arab media outlets to monitor, BMS provided a 24/7 eye on what was happening in the information space. Countless times, BMS captured a piece of news that our human monitors did not catch. With the information battle space a key component of war fighting, we had a responsibility to not only be first with our information, but also to ensure media reporting was accurate and provided the right context and characterization. BMS was a key enabler for the communication team.

The technical transfer and integration of operational engines from DARPA/GALE as described in Operational Integration of STT and MT for Rich Transcription and Translation of Speech (Chapters 1-3) became essential to demonstrate the value of research in responding to critical national security and military needs (Section 6.2.2).

On a yearly basis, TSWG and BBN bring together the users of the BMS and WMS. Lessons that have been learned from real users include:

It WORKS! It is an integral piece to our daily operations. It speeds intelligence action; it doesn't need systems personnel to work it. It is straightforward and intuitive. It only takes 1-2 hours to train someone to use it.

Positive comments do not come only from early adopters, but also from the latest deployments in Iraq. Starting in 2009, the users' meeting agenda included review of the Foreign Media Monitoring System funded by the Army Machine Foreign Language Translation Software (MFLTS) to gather feedback and determine new requirements. MFLTS acquired systems that fused together BMS and WMS capabilities. The Multi-National Coalition in Iraq called in to the 2009 users' meeting to share their feedback, stating:

Our Open Source Team fills in some expected changes in reporting that are a result of having U.S. forces out of the Iraqi cities since June. The reliability of Open Source reporting can be questionable so having multiple sources to review allows us an opportunity to get different perspectives on the same story—[the system] has given us eyes on events that we do not ordinarily have access to due to the June Security Agreement that removed U.S. Forces from the Iraqi cities. Immediate unfiltered access to reporting in local and national Iraqi media allows for immediate validation of other intelligence reporting, i.e., validate claims of attacks, confirm events.

The BMS has become an essential tool for intelligence analysts as showcased by Bemish (2008) and by user testimonials like those above.

6.2.1.9. Expanding the Impact of Automated Language Processing

From 2003 to 2009, TSWG and other IC and DoD groups have funded various versions of the BMS and WMS. All of these efforts have helped move language processing technology from the research lab to the hands of the warfighter. Like any extremely large organization, the DoD has its own procedures for deploying new systems and capabilities, ensuring that they are sustained in the field, that users are adequately trained, that system use is incorporated into doctrine, and that systems continue to be enriched to meet evolving needs.

Two DoD organizations have stepped in to organize the set of requirements coming from the field and to transform them into a cohesive program: 1) The Army Machine Foreign Language Translation Software (MFLTS) Program of Record has taken the lead in fusing the WMS and BMS into a single Foreign Media Monitoring (FMM) system. 2) The Defense Intelligence Agency (DIA) has taken the lead in supporting intelligence capabilities for the combatant commands at the joint level. These interagency efforts will benefit all the users of the deployed applications.

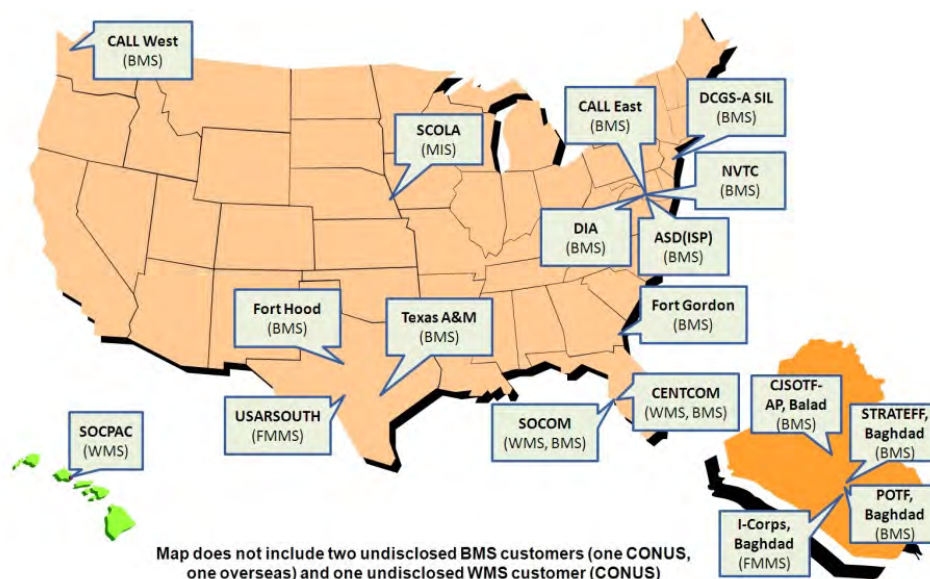


Figure 6.4: Current BBN Media Monitoring Deployments, January 2010

Starting with a few experimental efforts around the year 2000, the scope and maturity of deployed systems have increased significantly. Currently, there are 24 sites with 86 channels in the U.S. and overseas, with support in languages beyond the GALE core languages of Arabic and Chinese, including Farsi, Indonesian, Urdu, and Hindi. These language capabilities have been funded by TSWG, using GALE methodology to create and field operational systems. As these systems are put into service-wide use by the Army, DIA, and other organizations, language requirements will undoubtedly be addressed at the joint level.

These interagency efforts will benefit all the users of the deployed applications. New requirements are emerging to extend capabilities to more sources, such as radio, online video sharing sites, and social networks.

6.2.1.10. Dual Use of Technology

While the main users of the applications that have been described are language analysts and operators in the field, the BMS has been modified for dual use in the cultural awareness and language learning arenas. In every single operational deployment, language analysts have stated that they wished they had had these technologies available to them when they were learning a foreign language. Access to authentic materials, especially video content, is highly valued among instructors and language learners who aim to sustain their skills and stay up to date on cultural issues. In this role, the BMS provides instructors and students with access to authentic material that can be used in the creation of language-learning content and activities. This material is authentic, topical, and current, and contains pronunciations by a variety of native speakers. Accessing video cuts is a labor intensive task for instructors, and in the words of one of the instructors using the BMS: *The BMS saves me time and makes it easy to find the right videos to insert in my lessons.*

While analysts may need 24x7 monitoring of each source, the language learning community prefers to focus on key shows and times of the day to gather data of interest. This requirement led to the development of a DVR-like scheduler that an administrator can customize to capture a variety of programming around the clock, multiplexing many sources into a single BMS channel. This multiplexing capability reduces the overall system cost and exposes users to a greater variety of sources.

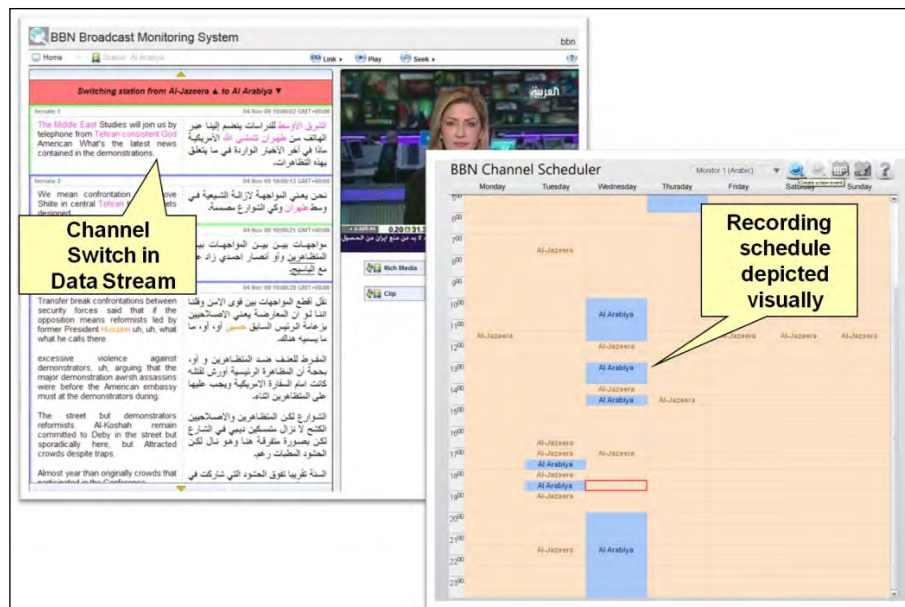


Figure 6.5: BMS scheduler

Three current deployments of the BMS to language learning centers provide instructors the ability to search video to create language activities based on authentic clips with full synchrony of video, audio, and text. This rich media is making it easy for teachers to create materials, and for learners to improve and sustain their language skills. In addition to making it possible for users to edit both the transcription and translation, third-party tools were integrated to enhance the learning process.

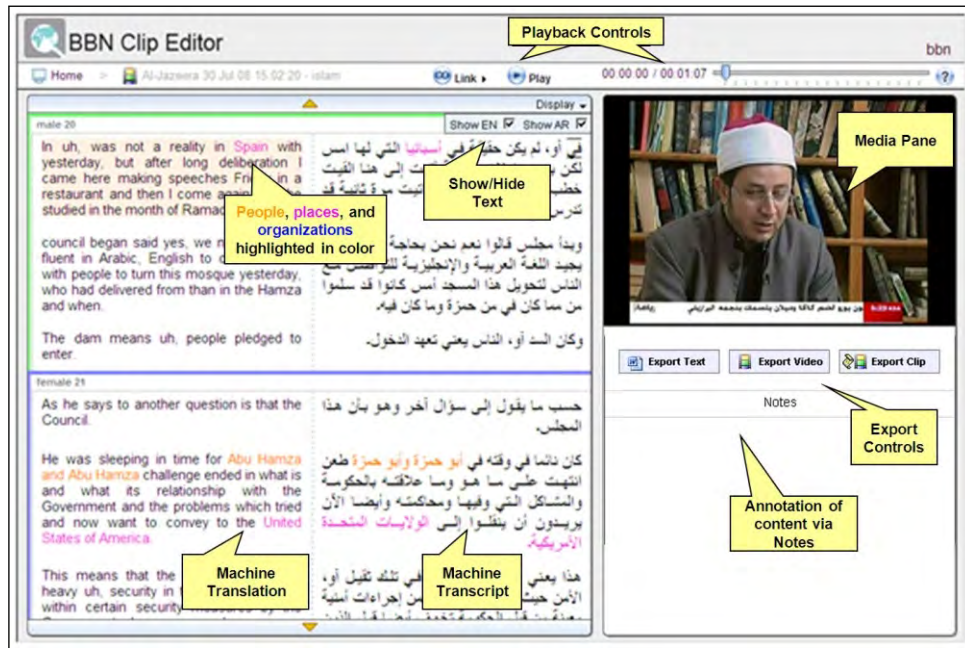


Figure 6.6: Clip Editor



Figure 6.7: Building Rapid Rate

The latest additional tool to be integrated provides the ability to create flashcards (for use in Rapid Rote from Transparent Language) with terms and expressions for further practice. Users can add notes to highlight cultural and linguistic features of a word, sentence and paragraph.

The adaptation of the BMS to language learning requirements has not only supported the needs of this new community of users, but also promoted development new functionalities and user interfaces that can be offered to all BMS users.

6.2.1.11. Conclusion

The improvements to operational engines made under the GALE program have made it feasible for the Combating Terrorism Technical Support Office/TSWG to insert state of the art human language technologies into capabilities that have been developed and deployed in operational settings. These technologies are not intended to be a replacement for human language analysts, but rather an extension to support their task. The heavy dependence of HLT training algorithms on language, training domain, sources, and channels is a problem that will need to be resolved in order to make fuller operational implementation of HLT possible. While machine-extracted content metadata hold great promise for effective data triage and operational adoption has been successful in assisting open-source analysis missions, world conditions and mission objectives are constantly changing, and the static nature of current HLT algorithms will need to evolve and adapt to the dynamic and unpredictable content of operational data. It is critical to emphasize the importance of human-machine collaboration in achieving successful deployment of HLT. An excellent approach is to focus on the tasks of the users and respond to requirements by soliciting the expertise of humans who engage the automated systems.

TSWG's mission is to facilitate the emergence of innovation based on user requirements and build effective solutions to support specific missions. TSWG will continue to rely on the research community to advance and improve the quality of automated speech-to-text and machine translation algorithms and expect to see those advances integrated into systems that fill operational requirements.

Addressing both ends of the research and development pipeline is vital to achieving success in providing functional solutions to the warfighter. For both DARPA/GALE and TSWG, success in bridging the gap between research and operational reality has been due to a combination of factors—vision, teamwork, listening to users, taking risks on new ways of doing business, and integrating top-of-the-line operational engines into user-friendly applications.

Acknowledgements

Since 2003, David Herrington of DoD/CTTSO/TSWG has been providing valuable resources and guidance. We also would like to acknowledge and thank all of the users - the early adopters- as well as the researchers and developers on the team who have made this DARPA/TSWG leveraging a success story. We would also like to thank Charles Wayne for his early contributions to the pre-GALE efforts.

6.2.2. Operational Integration of STT and MT for Rich Transcription and Translation of Speech

Authors: Daniel Kiecza, Amit Srivastava, Guruprasad Saikumar, Sean Colbath, Martha Lillie, Prem Natarajan, Abdessamad Echihabi, Daniel Marcu, and William Wong

6.2.2.1. Introduction

In this section, we describe the joint engineering effort between BBN Technologies and Language Weaver to produce a tighter and richer coupling between their respective Speech-to-Text (STT) and Machine Translation (MT) systems. The coupling includes the sharing of orthographic normalization rules, synchronization of lexicons between the two systems, and definition of an extensible interface that supports increased information flow between the engines. We show that tighter operational integration leads to better overall system performance. We demonstrate that the integrated rich transcription engine with automatic translation has great practical utility for intelligence analysis operations as the primary enabler of the BBN commercial off-the-shelf (COTS) Broadcast Monitoring System (BMS).

Goals

State-of-the-art STT capability at BBN can automatically transcribe foreign language speech at real-time throughput. The Language Weaver MT system offers automatic translation of text in many foreign languages into English. Both systems have been available as part of various commercial product solutions from BBN and Language Weaver, respectively, for some time.

The goal of the Operational Integration effort under the GALE program is to bring the two systems together into a joint tightly-integrated product offering that can produce a rich transcript of foreign language audio and video data with automatic translation of the transcribed speech content to English. The integrated system must run on a single commodity compute server and must be capable of continuous, real-time, low-latency processing of an audio signal.

Beyond the immediate benefits of an integrated operational system we anticipate that the integrated system should match or exceed overall system performance in comparison to operating a loosely-connected pipeline of the two individual systems.

Challenges

In our attempt to integrate the BBN Audio Monitoring Component (AMC) and Language Weaver MT systems we faced several formidable challenges.

BBN and Language Weaver (LW) have invested significant resources for many years to productize their respective systems. In the process, each system was tuned to perform optimally on state-of-the-art commodity hardware. Can the two systems run on a single server and still meet their respective operational requirements? Or do we need to scale each system down significantly to make the integrated system work and will this cause significant degradation in performance?

The BBN and LW systems have evolved independently, each defining its own input and output data formats without regard for the other. Naturally, these formats are not

compatible as is. Moreover, there are no available standardized representations for rich transcription output of STT or MT systems. Can we define a joint data format that supports the goal of the effort and that that can be made compatible with both systems?

Approach

Product-quality operational integration of STT and MT involves two major aspects: tight software-level system integration and data format specification, i.e. first integrate the systems into a single data pipeline through which data flows seamlessly, and then specify the data format in which the systems exchange information.

Our integration approach consisted of the following steps:

1. Identify and resolve significant incompatibilities between the systems, including mismatched orthographic normalization rules, unsynchronized system lexicons, and assumptions on acceptable durations of sentences.
2. The AMC product already emitted rich transcription according to a formal XML-based data format specification. We chose to leverage it for the development of an STT/MT information exchange format used to communicate between AMC and the MT system.
3. While the primary mode of using the MT system is via a SOAP interface, it also already supported other ways of integration. In particular, the MT system can be linked as a library into C++ applications. AMC was already a pipeline of C++ components, so we chose to integrate the MT system via this library. We developed a wrapper component to embed the MT system into the AMC pipeline as just another C++ application.
4. Testing of the integrated system on real world data such as live satellite television feeds for extended periods of time to ensure that the system operates stably and that it meets the functional requirements.

6.2.2.2. BBN COTS Rich Transcription System

The BBN Audio Monitoring Component (AMC), which is used for operational integration of STT and MT, is a software solution for creating rich transcriptions of speech in audio recordings. The transcript contains metadata beyond the raw STT annotation to enable improved human readability of the speech content. All elements in the transcript are indexed to the time at which they occur in the source audio stream. The following figure illustrates the richness of the AMC transcript: it is segmented into sentences and proper names are highlighted with color.

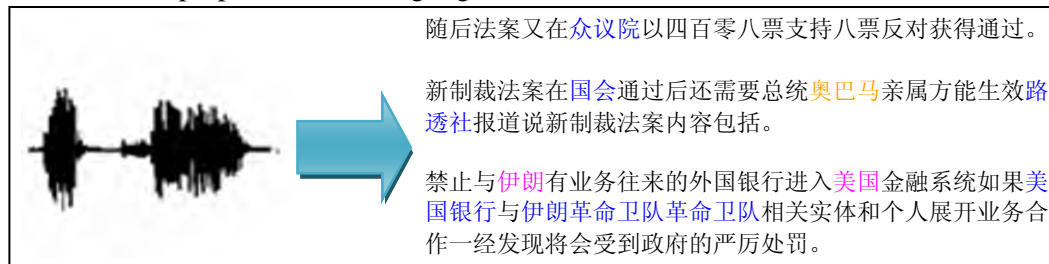


Figure 6.8: AMC – audio signal to rich transcription

System Description

The AMC integrates a suite of computationally demanding signal processing and pattern recognition techniques to produce a rich transcription of an audio input signal. The following figure shows an overview of the AMC system architecture.

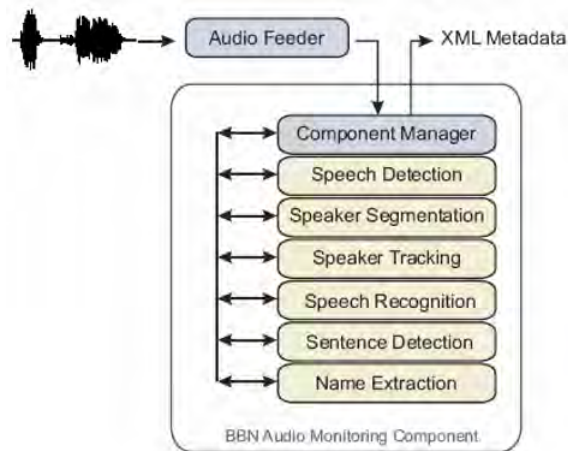


Figure 6.9: Integrated AMC component pipeline

The AMC components are arranged in a simple data pipeline. The final output of the AMC system is an XML-based transcript of the input audio signal. Included in the transcript are the following features:

1. Automatic word transcription
2. Named entities: people, locations, and organizations
3. Speaker turns and gender
4. Sentence boundaries
5. Non-speech audio events such as music or silence

At the time of this writing, the AMC product supports the following languages for rich transcription:

1. Modern Standard Arabic – with integrated Iraqi dialect
2. Mandarin Chinese
3. Persian/Farsi
4. Western Hemisphere Spanish
5. North American English

The system is tuned to perform at an average throughput time of $\frac{1}{2}$ -times real-time, so a one hour audio file is processed in approximately 30 minutes. The AMC provides an HTTP input and output interface. The system accepts audio data and returns the rich transcription and translation through this interface. The transcription and translation

output is represented using an XML data format. BBN has a fully documented API to support integration into other applications.

6.2.2.3. LW COTS Machine Translation System

This section describes the commercially available Language Weaver Machine Translation system used for operational integration of STT and MT. The Language Weaver COTS MT product is a statistical system that translates text from one language to another while preserving any phrase annotations.

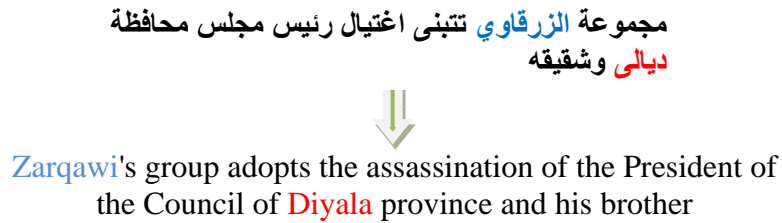


Figure 6.10: MT with linked name annotations.

The LW MT product was engineered to enable organizations to meet the volume, speed, and accuracy requirements for on-going translation demands. It is deployed in over 100 installations and has been integrated into over 20 complementary applications. Key features are:

- Client-Server architecture for on-premise, large-scale and distributed deployment
- Industry-standard Web services interface
- Flexible input and output formats
- MS Office support via Open Document Format
- Customizable terminology translation
- Phrase annotations linked across translation

The LW Machine translation system supports two translation strategies: phrase-based and syntax-based translation.

The phrase-based system is based on translating phrases that are extracted automatically from word-aligned parallel text. The system uses a log-linear combination of translation models and an n-gram language model. The weights of these models are discriminatively tuned on a held-out development set. The system requires 2GB of RAM and delivers translation speeds ranging from 1000 wpm to 5000 wpm depending on the desired quality.

The syntax-based system is based on tree transducer rules that are learned automatically from a set of tuples of the type (English parse trees; foreign sentences; word alignment). The system employs a CKY-style decoder that uses a log-linear combination of translation models and an n-gram language model. The system requires

4GB of RAM and delivers translation speeds ranging from 100 wpm to 1000 wpm depending on the desired quality.

6.2.2.4. Operational Integration

This section describes the technical details of integrating BBN AMC and Language Weaver MT.

Desired Capabilities

When this effort started, the AMC served already as the rich transcription engine in the BBN BMS. Our operational integration work was guided by the goal of enhancing the BMS with automatic translation of the transcribed foreign language speech into English. The integrated system should run on a single commodity compute server, support continuous, real-time, low-latency processing of audio data and operate stably for months at a time with no need for human maintenance.

The STT/MT interface and the associated data exchange format should provide the following capabilities:

- Structured representation: The data should be formatted as XML which is capable of encoding all the indexed features by STT, named-entity recognition, and MT. XML's Unicode support allows unambiguous encoding of source language and English translation text. In addition, the representation can be easily extended as needed.
- Sentence-by-sentence translation: AMC transcribes speech content in the form of sentences. The MT system is required to translate the source language transcript one sentence at a time.
- Crosslingual name linkage: AMC annotates categorized as *person*, *location*, and *organization* in each source language sentence. The MT system must preserve the identical set of names in the English translation. The data format must encode name annotations in both the source language and the English translation and it must preserve an unambiguous link between a source name and its translation.
- Output constraints specification: In order to improve human readability of the source transcript and its translation, the data format must define constraints on the lexical representation of both the source transcript and the translation, including controlling the range of legal characters, regulating usage of capitalization and punctuation, and removing typographical anomalies.

Integration Issues

This section summarizes the issues that we addressed to achieve effective operational integration of AMC and the LW MT system. The overarching theme of these issues is that the two systems were developed independently, and so various engineering design choices involved in developing each independent system led to inherent incompatibilities between the systems.

First of all, the systems were mismatched as to the domains in which they were designed to operate. AMC was optimized to handle audio signals that contain broadcast news recordings from television and radio sources, while the intended use of LW MT is to translate written text such as Web pages and documents. While there are various potential problems associated with this mismatch, the most important one turned out to be the length of a translation unit. Whereas LW MT was designed to perform optimally on sentence units as is typical for written text, but AMC was not yet capable of producing sentence-like STT output with average lengths similar to text, and a maximum length not to exceed a practical hard upper limit. AMC routinely emitted token sequences of lengths that would cause the MT throughput performance to degrade noticeably. To solve this issue, we developed and integrated a discriminative sentence boundary detection component into AMC that rendered the system capable of emitting well-formed sentence units. Details are provided in the article entitled *Discriminative Sentence Boundary Detection for Robust Operational Engines* in this chapter.

Differing system lexicons were another source of incompatibility between the systems. The LW Arabic-to-English MT lexicon was three times the size of the AMC Arabic STT lexicon; yet it covered only about 63% of the STT words. More than one third of all tokens emitted by the STT system were unknown to the MT system! We addressed this issue by first analyzing and categorizing the unknown words. While our analysis showed that a good portion of the unknown words consisted of proper names and of legal morphological variations, we discovered that a large proportion of these had one simple cause: the two systems used different sets of orthographic normalization rules during the lexicon design process. We carefully synchronized the rule sets and found that the MT lexicon coverage of the STT lexicon grew by an absolute amount of over 24% to a value of 87%.

A final source of incompatibility was found in the runtime characteristics of the two systems. Specifically, each system was tuned to make optimal use of the then-best-available commodity server. Co-locating the systems on a single server caused the joint system to run very slowly – it exhibited end-to-end latency peaks above 300 seconds. This was significantly larger than a maximum latency of 100 seconds that BBN requires of the AMC system. We resolved this problem by jointly tuning the runtime characteristics of the two systems.

Software Integration

A significant part of our effort to integrate the AMC and MT systems was spent on designing how the two systems would interoperate at a software component level. Originally, LW recommended integrating with its MT system via its SOAP I/O interface. While we considered this path for operational integration we ultimately selected a different approach: the MT system is also available as a C++ link-library. We, thus, integrated it into a C++ wrapper application to access a function-level interface to the MT engine. As shown in Figure 6.11 the wrapper application contains the MT system (gray box).

The C++-based approach proved desirable for a number of reasons. Firstly, the wrapper application was easy to develop using the AMC Component Manager (CM) client library and it yielded tight component integration into the AMC processing pipeline via function calls into the MT library. Secondly, this design placed the MT component

under system state management of the CM. By design, the CM integrates the individual component states into an overall AMC system state. Now, the AMC system state also reflected the state of the MT system. In particular, in the event that MT encountered a fatal error, the AMC system state could immediately reflect this information. Further, several AMC components require large in-memory models to operate, including the MT system. To minimize system startup time and to protect the system hard drive the CM starts the pipeline components in a staggered fashion. Under CM control the MT system automatically participated in the staggered startup procedure.

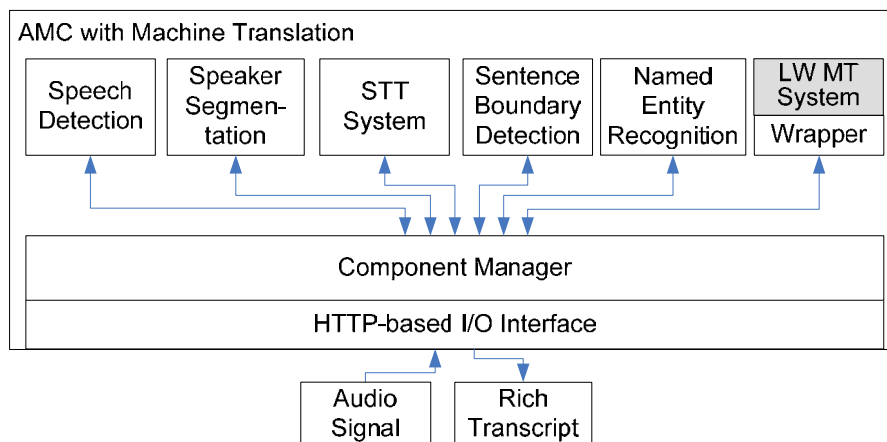


Figure 6.11: Component-level integration of AMC and MT

Data Format Specification

We have jointly developed a formal specification of the STT/MT data format. As described in the previous sections, this format is derived from the pre-existing specification of the AMC rich transcription format.

Specifically, the following features of the AMC rich transcription output are preserved in the STT/MT format:

- Lexeme tokens: The source language lexeme tokens of a sentence form the basis on which MT produces an English translation.
- Sentence boundaries: The MT system implicitly utilizes the sentence boundary information since translation is done one sentence at a time.
- Named entities: Named entity attributes including name boundaries, name types, and unique name id are provided as input to MT which, in turn, is required to preserve each name in the MT output.

Benefits of Integration

The joint BBN—Language Weaver effort under GALE succeeded in creating an operational system that tightly integrates the BBN AMC and LW MT systems. The resulting *AMC with MT* system produces a rich transcript of foreign language speech

contained in broadcast news recordings with automatic translations into English of the speech transcript.

AMC with MT provides several benefits over loosely combining the individual systems that were available before the integration effort. Firstly, as a result of our extensive synchronization work the translation performance of *AMC with MT* on speech data is demonstrably better than that of the loosely connected individual systems. Secondly, the integrated system produces an improved rich transcription that preserves linkage of names across translation. And finally, by design *AMC with MT* runs on a single commodity server resulting in better scalability. Prior to integration, each of the two individual systems required its own server for proper operation.

We have tested *AMC with MT* extensively to ensure runtime stability within the specified operational constraints for extended periods of time. Since then *AMC with MT* has been available as part of the BBN suite of commercial product offerings. In particular, *AMC with MT* is a critical enabler of one of BBN's flagship products: the Broadcast Monitoring System (BMS). The BMS creates a continuous searchable archive of international television broadcasts in real-time.

Finally and most importantly for the GALE program, after many operational deployments of systems and solutions that integrate *AMC with MT*, it is clear that a system such as this has become a much appreciated and essential tool for intelligence analysis operations on open-source broadcast media sources. In effect, the system enables English-only speaking analysts to reach across the language barrier and to monitor and gist broadcast media in languages that they themselves do not understand.

6.2.2.5. Performance Evaluation: the BBN AMC, LW MT, and Integrated STT/MT

This section describes the STT performance of BBN AMC, the MT performance of LW MT and the performance of the integrated STT/MT system over the course of the GALE program.

Technology Integration

STT Technology Integration	
1.	Heteroscedastic Linear Discriminant Analysis
2.	Perceptual Linear Predictive (PLP) features
3.	LM interpolation and new backoff estimation
4.	Discriminative STT training
5.	Log-domain probability computation
6.	Support for large corpus training – Arabic: 1000+ hours, Chinese: 500 hours
7.	Discriminative named entity recognition
8.	Perceptron-based sentence boundary detection

Table 6.1: Technology integration into BBN AMC

MT Technology Integration	
1.	5-gram language models
2.	Language models trained on very large corpora
3.	Better rule-based components
4.	More accurate word alignment models
5.	Better feature weights tuning
6.	Improved translation models
7.	Syntax-based MT models

Table 6.2: Technology integration into LW MT

One goal of the GALE program is the transfer of proven research advancements into systems that can be deployed for operational use. Largely independent of each other, BBN and LW continue to integrate new research improvements into their operational systems. Table 6.1 summarizes the specific research improvements that have been integrated into BBN AMC during GALE.

Table 6.2 summarizes the specific research improvements that have been integrated into the LW MT system during GALE.

AMC System Performance

Figure 6.12 tracks the performance of AMC Arabic during the first three years of GALE. Performance is measured using the Word Error Rate (WER) metric on the GALE-2006 speech development sets – split into the broadcast conversation (BC) and broadcast news (BN) portions of the set.

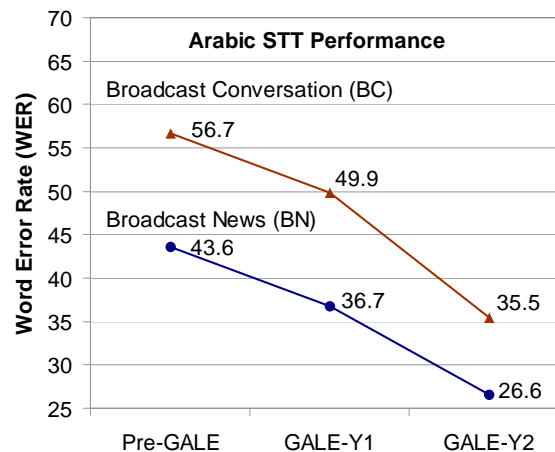


Figure 6.12: Arabic STT performance during GALE

Figure 6.13 tracks the performance of AMC Chinese over the course of the first three years of the GALE program. Performance is measured on the GALE-2006 speech development sets – broken up into the broadcast conversation and broadcast news portions of the set. The GALE-improved Chinese system achieves a dramatic 50% relative reduction in WER on the Chinese conversational data compared to the pre-GALE

AMC system. We have achieved a remarkably consistent and large WER reduction for both Arabic and Chinese.

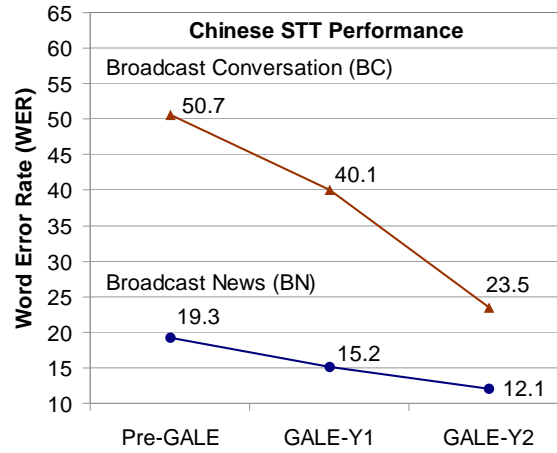


Figure 6.13: Chinese STT performance during GALE

MT System Performance

Figure 6.14 tracks the performance of the phrase-based Arabic-to-English MT system during the first three years of GALE. Performance is measured using the Bilingual Evaluation Understudy (BLEU) metric on a 1,000-sentence newswire test set with 4 translation references.

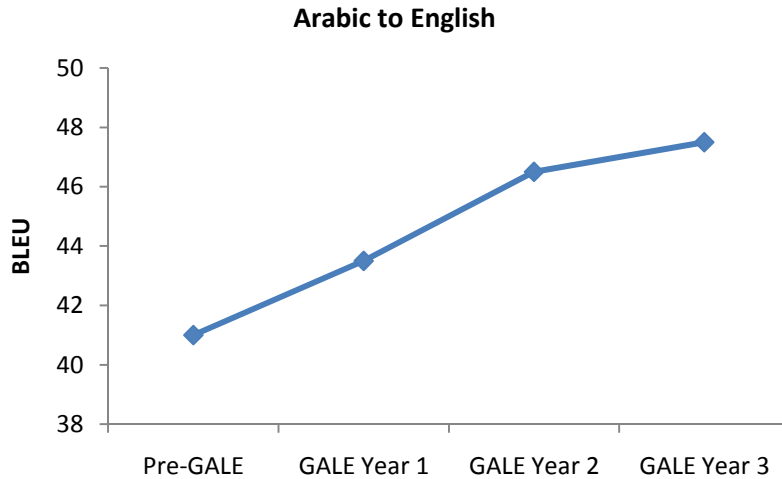


Figure 6.14: Arabic-to-English phrase-based translation performance during GALE

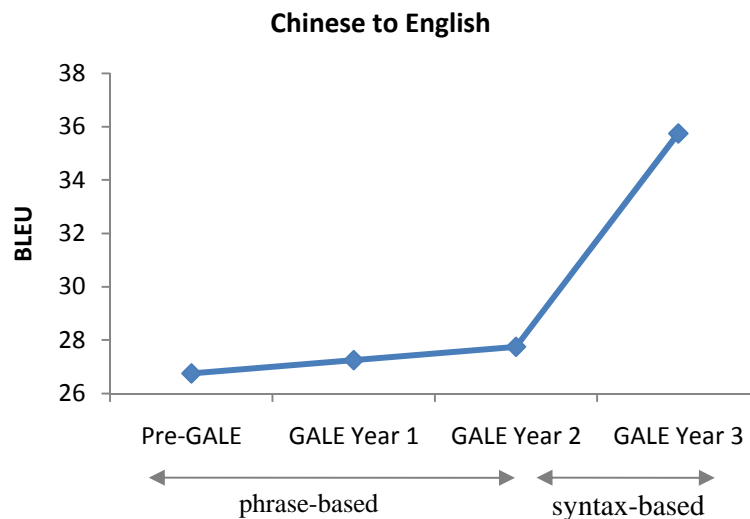


Figure 6.15: Chinese-to-English translation performance during GALE

Figure 6.15 tracks the performance of the Chinese-to-English MT system during the first three years of GALE. Note that LW introduced Syntax-based MT in the product. Performance is measured using the BLEU metric on a 1,000-sentence newswire test set with 4 translation references.

AMC-with-MT System Performance

In addition to tracking performance improvements in the individual engines we also track the cumulative effect – translation performance of LW MT on AMC STT output. We score the *MT on speech* performance using the Translation Error Rate (TER) metric with a single reference and with mixed case and punctuation. Performance is measured on the GALE-2006 speech development sets. All results are produced by COTS engines in standard configuration. In particular, all systems run at real-time with an average end-to-end latency of less than one minute. Figure 6.16 tracks the translation performance of the integrated system on Arabic speech data during GALE.

Figure 6.17 tracks the translation performance of the integrated system on Chinese speech data during GALE.

As the figures demonstrate, we have achieved consistent performance gains across languages and genres. The translation performance on speech data of the GALE-enabled integrated system has improved significantly over the pre-GALE performance baseline.

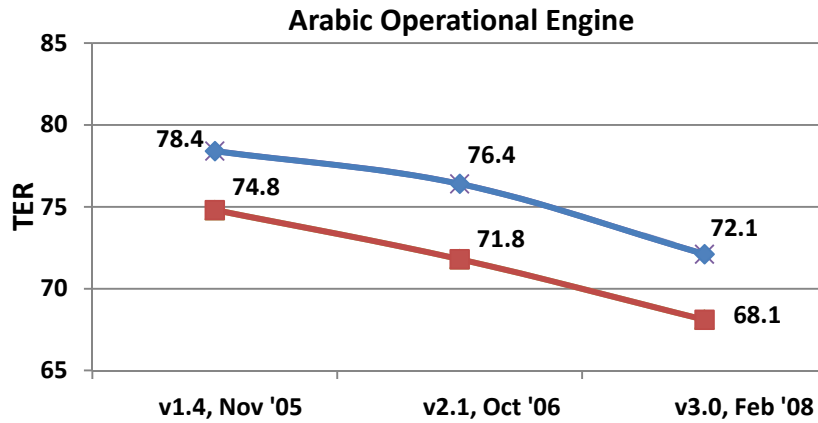


Figure 6.16: MT performance on Arabic speech during GALE

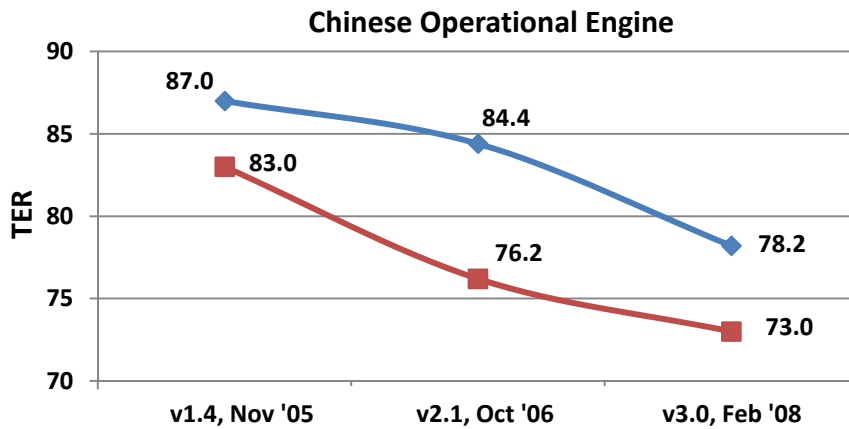


Figure 6.17: MT performance on Arabic speech during GALE

6.2.2.6. The BBN Broadcast Monitoring System

BBN's commercial Broadcast Monitoring System (BMS) was developed in partnership with the Technical Support Working Group (TSWG), a research division under the Assistant Secretary of Defense for Special Operations/Low Intensity Conflict. TSWG focuses on rapid prototyping of new solutions with the ultimate goal of transitioning robust products to end users. In late 2001 when TSWG was keen on providing soldiers with the capability to access non-English media, specifically international television, they surveyed the technological landscape and selected BBN's speech and language processing technologies (Shepard 2002). Since its inception, the BMS has integrated the immense advances accomplished under DARPA-funded research efforts and in particular the current GALE effort.

The BMS is a turnkey COTS product (hardware and software) that creates a continuous searchable archive of international television broadcasts. Real-time broadcast streams from satellite or cable are captured by the system and automatically maintained in a one-year archive. As video is ingested by the BMS, the audio stream is automatically transcribed by the AMC and translated into English by the Language Weaver MT system. The browser-based interface on the BMS lets users access the BMS from any PC or laptop via a common high-speed IP network. The only required client software is Microsoft® Internet Explorer and Windows Media® Player.

The following figure shows the home screen of the BMS – the Channel Overview. It contains a search box, a list of persistent queries (the watchlist), and a thumbnail overview of the available channels.



Figure 6.18: BBN BMS channel overview

The BMS enables its users to quickly find specific spoken content in the video archive through keyword queries — in English or the source language — and to jump directly to the streaming video from the search results. Searches can be scoped by channel, language, and time.

Figure 6.19 shows the BMS Channel View, which displays the rich transcript and automatic translation side-by-side with the original video signal. The video can be accessed and played back from any point in the archive by clicking on a word in the transcript. As video plays back, the spoken words and translation are highlighted. The transcript and translation are enriched by highlighting the names of people, places, and organizations in color, and by separating the speakers in the audio file. This integrated view helps users to converge on the meaning of the broadcast.



Figure 6.19: BMS channel view

Users can save collections of searches in a watchlist that continuously monitors incoming video for matches to search criteria. Video segments are automatically added to the watchlist as they are broadcast and the system alerts the user as new matches occur. Watchlists can be filtered by date, user, and topic, so users view only watchlist items of interest to them.

Users can extract video segments or still photos for collaboration, presentation, and reports. Rich media — integrated video, transcript, and translation packages — can be exported to HTML files for playback in a browser or to BBN Translator’s Aide, an application that improves the efficiency of human translation from video sources.

The BMS is available as a turnkey solution, requiring no hardware setup or software installation by the customer. The system is designed to run as an appliance without needing any onsite system administration. The BMS turnkey solution is designed to comply with the Defense Information Systems Agency (DISA) Gold standard to support deployment security requirements. The BMS has been deployed operationally since August 2004 for open source intelligence collection and information operations at many locations around the world.

The groundbreaking capability offered by the BMS enables users with no foreign language skills to get the gist of international television and radio broadcasts and to triage enormous volumes of media, allowing skilled linguists to focus on translation tasks. Realizing this capability required both fundamental research in human language technology, provided by DARPA, along with applied technology transition and user focus, provided by TSWG. For more information, see Section 6.2.1. More information on the BMS is available on the BBN website.¹

¹ http://www.bbn.com/products_and_services/bbn_broadcast_monitoring_system

6.2.2.7. Conclusion

With the commercial availability of AMC with MT, for example as part of the BBN Broadcast Monitoring System product, we have achieved the goal of this effort to demonstrate a practical integrated system that creates a rich transcript of foreign language audio data with automatic translation into English. The MT performance on speech is significantly improved over the pre-GALE system which consisted of a loose connection of the individual systems. AMC with MT also provides additional metadata: annotated and linked name phrases in both the source language and the corresponding English transcript.

Over the course of the GALE program, we have continually transferred practical research advancements into both AMC and the LW MT system and are, thus, able to offer the resulting performance improvements in our deployable systems. In fact, we managed to demonstrate that cutting edge research advancements achieved during GALE can be transitioned into deployable systems in as little as 12 to 18 months.

We continue to transition our systems into operational deployments and we have found that these systems are rapidly becoming a much appreciated and essential tool to assist intelligence analysis operations on open-source broadcast media sources. In effect, the systems enable English-only speaking analysts to reach across the language barrier and to monitor and gist broadcast media in languages that they themselves do not understand.

6.2.3. Discriminative Sentence Boundary Detection for Robust Operational Engines

Authors: Amit Srivastava and Daniel Kiecza

6.2.3.1. Introduction

We present a discriminative Sentence Boundary Detection (SBD) component for speech that not only improves the accuracy of sentence segmentation; it also improves the overall robustness of the system, enabling real-world deployments of GALE operational engines. In this section, we describe the combination of two SBD subsystems in the BBN operational engine: a discriminative Neural Network (NN) model built over prosodic features estimated from the underlying speech signal, and a Perceptron model that uses linguistic and prosodic information jointly. We also describe the analytical model used to estimate a length-based penalty during decoding to constrain the output sentence lengths. The discriminative SBD component outperforms the pre-GALE SBD component in both, the sentence segmentation performance on Arabic and Chinese speech, as well as the end-to-end system latency of the operational engine when processing live audio.

As speech recognition, machine translation, and natural language processing technologies mature, there is an urgent need to transition the proven algorithms and techniques into the field to provide tools to analysts for effective distillation of information from large volumes of foreign speech. In addition to integrating and demonstrating the latest research algorithms and models, the GALE operational engines

need to demonstrate a practical capability that can do useful work in the real world. In particular, we defined the core capabilities of the integrated operational engine as follows. The system would have to:

- Run integrated speech-to-text (STT) and machine translation (MT) on a single commodity compute server to produce sentence-based translation of the input foreign speech,
- Support continuous, real-time,² low-latency processing of audio data and,
- Operate stably for months at a time with no need for human maintenance

The BBN Audio Monitoring Component (AMC) is an example of a practical, integrated operational engine that produces automatically and in real-time a rich transcript of foreign language speech. Sentence boundaries are important structural metadata that enhance speech recognition transcripts in terms of human readability and natural-language processing. Parsers, information extraction systems, and machine translation systems rely on sentential inputs from speech-to-text systems for optimal performance. The Sentence Boundary Detection (SBD) component in the AMC is critical to providing high-level linguistic structure in the output rich transcription, enhancing the accuracy and readability of the English translations, and enabling operational viability of the integrated operational engine on real world data in the field.

Most SBD systems are statistical engines that learn linguistic and acoustic cues from human-annotated data to hypothesize sentence boundaries in speech transcripts. Limitations in the SBD algorithms and models as well as unspecified constraints can lead to very long sentence hypotheses that have many adverse effects on the operational engines:

- Degraded readability and comprehension – long sentences are difficult to parse and understand especially in the presence of speech-to-text and machine translation errors
- Inefficient MT – long sentences increase the potential search space for statistical machine translation engines leading to reduced speed and accuracy of translations
- Increased end-to-end system latency – in pipelined, integrated operational engines that produce sentence-based translations of the input foreign speech, such as the BBN AMC, the length of the sentence directly influences the system latency and hence the practical usability of the system in the field.

Previous research on SBD has focused on combining lexical and prosodic cues in separate models or in a single model-based framework to hypothesize sentence boundaries between words in the STT output (Stolcke and Shriberg 1996 and Liu *et al.* 2004). Model combination has been shown to significantly improve segmentation

² We use the term real-time to denote a throughput data rate that is the same or faster than the data rate of a live audio source.

performance over individual model outputs as in our pre-GALE approach (Stolcke *et al.* 1998). However, most of these approaches lack sentence-length constraints unlike the SBD approach described in this section, which has been developed specifically for integration in the BBN AMC to enable sentential segmentation in high-throughput, low-latency speech transcription for readability and translation.

The approach described in by Matusov *et al.* (2006) focused on improving MT performance from speech and included a sentence-length constraint similar to our approach. However the decoding algorithm in their approach uses an HMM-style search over the complete document whereas our approach uses a tagging framework to reduce end-to-end system latency.

6.2.3.2. Previous Approach and Limitations

Prior to GALE, the operational engine included a SBD system (Srivastava and Kubala 2003) that consisted of a combination of two subsystems: an Acoustic subsystem that consisted of a discriminative Neural Network model over prosodic features, and a linguistic subsystem that consisted of a generative N-gram language model over word-based trigram features. The scores from the two subsystems were combined in a Maximum-Likelihood framework in a Viterbi decoder to hypothesize sentence boundaries between words from each automatically segmented speaker-turn in speech.

Prosodic Feature Extraction

Figure 6.20 shows the conceptual diagram of our feature extraction method. We define putative boundaries (PB) as the intervals between words in the speech transcripts. The SBD system can hypothesize sentence boundaries in each of these PB. We extract a variety of prosodic features from the audio at each PB.

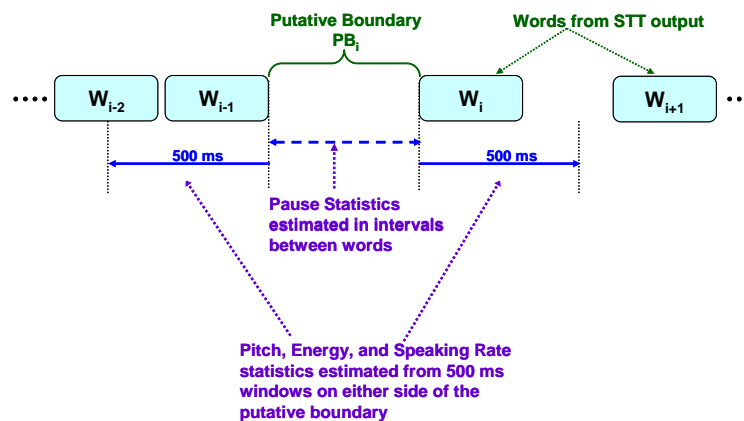


Figure 6.20: Prosodic feature extraction method

Pause statistics are estimated from the intervals between the words, which may be absent altogether when there is no pause between the hypothesized words. Pitch, Energy, and Speaking Rate statistics are estimated from 500 ms windows on either side of PB_i

extending from its edges. The estimated prosodic features are functions of these statistics. We used these four fundamental prosodic attributes to generate a total of 47 features.

Table 6.3 shows the distribution of the prosodic features estimated, as well as, the types of statistics used to estimate these features. There were 9 Pause related features, 2 Speaking Rate features, 6 Energy based features, and 30 Pitch related features estimated at the putative boundaries. The prosodic features estimated during this effort were a diverse set of features with some being continuous, others being discrete, and yet others were Boolean features.

Feature Type	Feature Description	#Features
Pause	<ul style="list-style-type: none"> •Pause Duration •Pause Attribute (Filler, Breath, etc.) •Time since last Pause •Normalized Pause Duration 	9
Speaking Rate	<ul style="list-style-type: none"> •Absolute Value •Difference across Putative Boundary 	2
Energy	<ul style="list-style-type: none"> •Absolute Values •Difference across Putative Boundary •First Difference of Energy 	6
Pitch	<ul style="list-style-type: none"> •Discontinuous Chains in Voiced Regions •Interpolated Continuous Pitch •First-Order Pitch Differences •Tone Estimates (Legendre Polynomials) 	30

Table 6.3: Distribution of prosodic feature attributes

Acoustic Subsystem

The NN model consists of a 2-layer feed-forward Neural Network that was trained on the 47 prosodic features described previously. Standard back-propagation training was applied with the Minimum Squared-Error (MSE) criterion to the NN with configuration: 47 input nodes, 4000 hidden nodes, and one output node (Srivastava and Kubala 2003). Sentence boundary classes are hypothesized by comparing the NN output score to a threshold as shown in the equation below. The output of the NN can also be viewed as the MSE estimate of the posterior probability of the class, sentence boundary. We will use this property of the Neural Network during model combination.

$$c_i = \begin{cases} \textit{sentence - boundary} & \text{if } \varphi(\tilde{F}_i) \geq \tau \\ \textit{non - boundary} & \text{if } \varphi(\tilde{F}_i) < \tau \end{cases} \quad (6.1)$$

Here, $\varphi(\tilde{F}_i)$ denotes the Neural Network output when presented with the input 47-dimensional *prosodic* feature vector \tilde{F}_i , and τ is an empirical threshold.

Linguistic Subsystem

For the Linguistic subsystem, we inserted the *sentence-boundary class* as the token, $\langle s \rangle$, in the word sequence for the transcripts in the training data and estimated a trigram language model (LM) using the BBN ByblosTM Speech Recognition LM toolkit. The resulting LM is very similar to the hidden segment model proposed by Stolcke and Shriberg (1996). Viterbi decoding is used to find the most likely class sequence given a sequence of words and the trained language model. The most likely class sequence is estimated as:

$$\hat{c}_{[1:n]} = \operatorname{argmax}_{c_{[1:n]}} \sum_{i=1}^n \log(\mathbf{p}^L(w_i / w_{i-1}, w_{i-2}, c_{i-1}, c_{i-2})) \quad (6.2)$$

where \mathbf{p}^L denotes the trigram Language Model probability for the context that includes the current word w_i and two preceding tokens from the set $\{w_{i-1}, w_{i-2}, c_{i-1}, c_{i-2}\}$.

ML Combination Framework

In the *Acoustic* subsystem, the NN produces scores, $\varphi(c_i / \tilde{F}_i)$ that are MSE estimates of the posterior probability of the boundary classes. These scores are transformed into likelihoods by scaling them with the prior probability, $P(c_i)$, of the boundary classes, estimated from the same training data.

The *Combined* system computes an exponentially-weighted product of the likelihoods from the two subsystems in a Maximum-Likelihood framework. The optimal sequence of boundary classes \hat{c}_1^n , given the sequence of words w_1^n , and the sequence of prosodic feature vectors \tilde{F}_1^n in a speaker turn with n words, is determined using the Viterbi algorithm as follows:

$$\hat{c}_{[1:n]} = \operatorname{argmax}_{c_{[1:n]}} \sum_{i=1}^n \lambda \log\left(\frac{\varphi(c_i / \tilde{F}_i)}{P(c_i)}\right) + \delta \log\left(\mathbf{p}^L(w_{i+1} / w_i, w_{i-1}, c_i, c_{i-1})\right) \quad (6.3)$$

Powell's minimization technique was used to search for the optimal weights, λ and δ that correspond to the minimum SBD error over the development test set.

Impact on Sentence Length

The Viterbi decoding framework needs well defined end-points for optimal search. Automatic speaker turns hypothesized by the speaker segmentation component in the AMC (Liu and Kubala 1999) provide these end-points for sentential decoding. However, speaker turns in speech can be very long. In the above approach it is very difficult to integrate sentence length constraints. This effectively implies that a sentence could span the entire speaker turn, which can lead to adverse effects on the operational engine.

In the next section we will describe the new formulation for SBD that will allow more diverse and discriminative features to be integrated into a statistical modeling framework and also allow sentence-length constraints to directly influence the placement of the boundaries.

6.2.3.3. Discriminative Tagging Framework

We formulate the Sentence Boundary Detection task as a tagging problem where the goal is to tag each word in the hypothesized transcription with one of the two classes: *sentence-boundary* (SB) class, and the *non-boundary* (NB) class. The hypothesized transcription is produced by the automatic speech recognition engine in the AMC. The sentence boundary tags are hypothesized independently at each word boundary by examining a fixed context around the word in each hypothesized speaker turn.

Approach

The SBD component in the AMC consists of an incremental decoder that accumulates speech utterances from the output of the Speech Recognition system in an input buffer. Each utterance consists of acoustic features, the top-1 word and phonetic transcripts. Acoustic and Indicator-function features are estimated and used by the Acoustic and Perceptron subsystems to produce scores for the two sentence classes: sentence-boundary and non-boundary. These two scores are combined with a sentence-length-based penalty score based on the comparison of the current sentence length with the average sentence length for each language. A sentence boundary is hypothesized at the putative boundary if the score for the sentence-boundary class is greater than the score for the non-boundary class. At this point, all but the last sentence are output from the AMC and the last sentence is retained in the input buffer for the next pass of decoding.

Feature Extraction

Features for sentence boundary detection are functional representations of eight fundamental attributes of the transcription and the underlying speech signal. There are four linguistic attributes: the word identity, the POS tag associated with the word, the word prefixes, and the word suffixes. There are also four prosodic attributes: pause, pitch, energy, and speaking rate. We used these attributes to create two types of features: Prosodic and Indicator-function. The prosodic features, described in detail in Section 2.1, were used in a NN model as in the previous approach.

Indicator-function Features

Attribute Type	Symbol	Attribute Values
Linguistic	W	<ul style="list-style-type: none"> • word identity • word suffices up to length 4 • word prefixes up to length 4
Linguistic	P	associated POS tag
Prosodic	F	<ul style="list-style-type: none"> • Pause duration quantized to 0.5 secs

Table 6.4: Attributes used to define history at a word

Similar to the formulation of Huang and Zweig (2002), we denote H as the set of all possible contexts, or *histories*, of the word that is to be tagged, and C as the set of all possible classes or tags. In our case, the size of the set C is 2. The transformation function

Φ , where $\Phi: H \times C \rightarrow R^M$ maps a history-class pair to an M -dimensional feature vector. Each component of the feature vector is represented as $\Phi_s(h_i, c_i)$ and denotes a particular indicator function over the history-class pair (h_i, c_i) at the word w_i .

We defined the history over a context of 5 words around the word that is to be tagged. The history includes all four linguistic attributes and one prosodic attribute. These attributes are defined in Table 6.4. The history h_i at a word w_i is then defined as:

$$h_i = \langle P_{i-2}, F_{i-2}, W_{i-2}, \\ P_{i-1}, F_{i-1}, W_{i-1}, \\ P_i, F_i, W_i, \\ P_{i+1}, F_{i+1}, W_{i+1}, \\ P_{i+2}, F_{i+2}, W_{i+2} \rangle \quad (6.4)$$

These indicator-function features encode co-occurrences of histories and classes. Effective features can denote that certain histories are strong predictors of the presence or absence of a sentence boundary class at a particular word in the transcription.

Perceptron Subsystem

The Perceptron model is a linear-discriminative model over the *indicator-function* features trained with the Averaged Perceptron algorithm (Collins 2002). The score of each boundary class from the Perceptron model at each word, given a fixed context around it, is given by:

$$S(c_i) = \sum_{s=1}^M (\alpha_s \Phi_s(h_i, c_i)) \quad (6.5)$$

Here, α_s denotes the weight attributed to the indicator function Φ_s . Thus, given parameter values, α_s and a history h_i in the context of word w_i , the highest scoring sentence class can be determined using the score from Equation 6.5. Training the Perceptron model consists of hypothesizing the sentence class at each putative boundary using the current parameter values α_s , followed by simple additive updates to α_s based on the true sentence classes, as explained in detail by Collins (2002).

Combined Decoder with Sentence Length Constraint

The scores from the Acoustic subsystem and the Perceptron subsystem are combined linearly with a sentence length penalty that discourages the combined decoder from hypothesizing sentence boundaries resulting in output sentence lengths that are shorter or longer than a target sentence length. An analytical sentence duration model is used to calculate the penalty score as a function of the difference in hypothesized and target sentence lengths as shown in the equation below.

$$P^D(SB/d) = \frac{1}{1 + e^{(d-d_t)}} \\ P^D(NB/d) = 1.0 - P^D(SB/d) \quad (6.6)$$

where, \bar{d} is the target sentence length estimated simply as the average sentence length from SBD training data for each language.

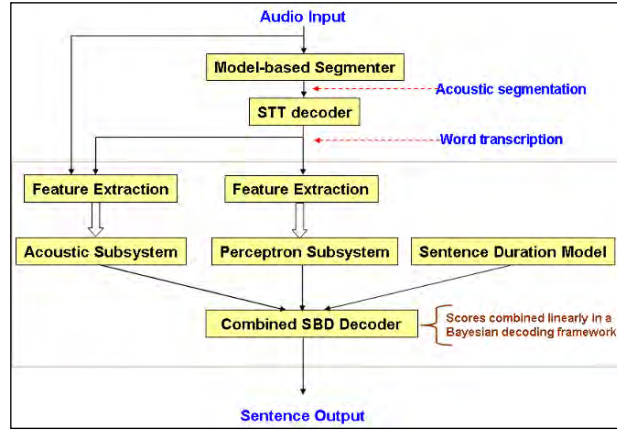


Figure 6.21: Sentence boundary detection in the BBN AMC

The Combined decoder, in Figure 6.21, computes the score for each sentence class at each putative boundary i using the equation:

$$\tilde{S}(c_i) = \mu(\sum_{s=1}^M (\alpha_s \Phi_s(\mathbf{h}_i, c_i))) + \theta \log \left(\frac{\varphi(c_i/\tilde{F}_i)}{P(c_i)} \right) + \gamma \log \left(p^D(c_i/\bar{d}) \right) \quad (6.7)$$

where:

- $\varphi(c_i/\tilde{F}_i)$ is the Neural Network output for sentence class c_i given the feature vector \tilde{F}_i
- 'd' represents the current sentence length given current position i and the last hypothesized sentence boundary in the turn
- $P(c_i)$ is the prior probability of sentence class c_i
- μ , θ , and γ are empirical weights that are estimated by tuning on the development test data

Sentence boundaries are then hypothesized using the Bayesian framework:

$$c_i = \begin{cases} SB, & \text{if } (\tilde{S}_i(SB) \geq \tilde{S}_i(NB)) \\ NB, & \text{otherwise} \end{cases} \quad (6.8)$$

6.2.3.4. Performance Evaluation

Arabic Experimental Data

We used Arabic Broadcast News (BN) acoustic data consisting of approximately 103 hours of spoken modern-standard Arabic, transcribed without diacritic markings, for our

experiments (Billa *et al.* 2002). This data is drawn from 10 different sources, which include Egyptian, Syrian, and Saudi broadcast radio and TV, and the Al Jazeera TV network. The transcriptions in this corpus are annotated with sentence boundaries. Approximately 96 hours of audio and transcriptions were drawn from this corpus to comprise the acoustic training set. We also used text articles drawn from Al Jazeera online website as additional text training data.

Set	Duration (hours)	# Words	# Sentences	%WER
Acoustic Training	96	600K	20K	-
Text Training	-	370K	21K	-
Dev	12	52K	2018	17.0
Eval	11	47K	1911	19.4

Table 6.5: Arabic experimental data summary

The remaining seven hours of audio from the Arabic BN corpus were divided into two sets, the SBD development (Dev) set and the SBD evaluation (Eval) set. Approximately sixteen hours of Arabic test data, both development and evaluation sets, from the DARPA EARS Rich Transcription Evaluations in 2003 and 2004 were manually transcribed with sentence boundaries at BBN and added to the SBD Dev and Eval sets respectively. All the data used in our experiments is summarized in Table 6.5. The AMC Speech-to-Text performance, in terms of Word Error Rate (WER), on the Arabic speech in the two test data sets is also shown in this table.

Chinese Experimental Data

Training data for Chinese SBD consisted of approximately 50 hours of speech transcriptions from the FBIS and LDC STT training corpus that contains manual annotation of sentence boundaries and punctuation.

The development and evaluation test sets were created with data drawn from five separate test data collections.

1. LDC Mandarin Development set for BN Hub4 Speech Recognition Evaluation 1997.
2. LDC Mandarin Evaluation set for BN Hub4 Speech Recognition Evaluation 1997.
3. LDC Mandarin Evaluation set for BN Hub4 Speech Recognition Evaluation 1998.
4. Mandarin EARS RT03+RT04 Development test set.
5. Mandarin EARS RT03+RT04 Evaluation test set.

Sets 1 to 3 contain manually annotated sentence boundaries from LDC while some files in sets 4 and 5 were manually annotated with sentence boundaries at BBN. Files without manual annotation in sets 4 and 5 were excluded from the development and evaluation test sets. Table 6.6 shows the composition of the data sets used for Chinese SBD experiments. The AMC STT performance, in terms of Character Error Rate (CER), is also shown for the test data sets.

Set	Duration (hours)	# Words	# Sentences	%CER
Acoustic Training	50	420K	24K	-
Dev	4	41139	1362	12.5
Eval	5	50863	1673	14.5

Table 6.6: Chinese experimental data summary

Evaluation Metric

The reference and the hypothesized Rich Transcription are represented as an ordered sequence of generalized tokens that includes the word identity and the sentence boundary information. The two sequences of generalized tokens are aligned with respect to a generalized edit-distance function that preserves the Speech-to-Text (STT) Word Error Rate (WER) while optimizing the SBD error. Sentence Boundary Detection performance is computed from this alignment by examining the sentence boundary attributes of each pair of aligned tokens. We used the Sentence Boundary Detection Slot Error Rate (SER) in the equation below as our primary evaluation metric (Makhoul *et al.* 1999). For all our experiments, we will present the system SER and the *Correct Rate*, which is 1.0 minus the *Deletion Rate*.

$$\text{SER} = \frac{\# \text{Insertion} + \# \text{Deletions}}{\# \text{References}} \quad (6.9)$$

Results

For each of the two languages, Arabic and Chinese, we evaluated three SBD systems: the Pre-GALE SBD system, the new discriminative SB tagging system without length constraints, and finally the new discriminative SB tagging system with length constraints. The empirical weights used in the combination framework were estimated using the Powell's method to minimize the SBD SER on the development test set. The optimal weights were then used to hypothesize sentence boundaries in the speech transcripts of the evaluation test set. The SBD performance on the Arabic and Chinese Dev and Eval sets is detailed in Table 6.7 and Table 6.8.

These results show that the new discriminative SBD system with local Bayesian decoding outperforms the Pre-GALE SBD system. There is a significant improvement in the correct percentage as well as in the SER performance for the new approach even with the restricted context imposed by the local Bayesian decoding framework. We expected the performance to degrade with the addition of the length constraints. However, as seen from the result tables, there is an insignificant change in performance for Arabic SBD even when the length-based penalty is added to the decoding equation although, surprisingly, Chinese SBD shows a modest improvement. We suspect that this improvement is related to the property of the Chinese language whereby Chinese sentences are much more well-defined structurally and are also much shorter, in average, than Arabic sentences. Imposing a sentence-penalty then that is influenced by the average sentence length turns out to be much more beneficial for Chinese than for Arabic.

System	Arabic Dev		Arabic Eval	
	% Corr.	% SER	%Corr.	% SER
Pre-GALE	75.6	53.6	75.4	49.3
Discriminative SBD without length constraints	79.6	50.8	79.8	46.9
Discriminative SBD with length constraints	79.9	51.4	80.3	46.0

Table 6.7: Arabic SBD results

System	Chinese Dev		Chinese Eval	
	% Corr.	% SER	%Corr.	% SER
Pre-GALE	70.3	48.4	66.5	59.7
Discriminative SBD without length constraints	76.0	42.5	71.9	56.7
Discriminative SBD with length constraints	79.4	44.5	76.6	54.6

Table 6.8: Chinese SBD results

Engine Runtime Performance

We compared the end-to-end system latency of the AMC engine with respect to the pre-GALE SBD and the new discriminative SBD systems. In both cases, the AMC was run continuously for five days on live audio feed captured from the Al Jazeera Arabic television channel via satellite. The end-to-end system latency is defined as the lag (in seconds) between the time at which a sentence is output from the AMC, and the time at which the audio corresponding to the beginning of the sentence was received at the input of the AMC.

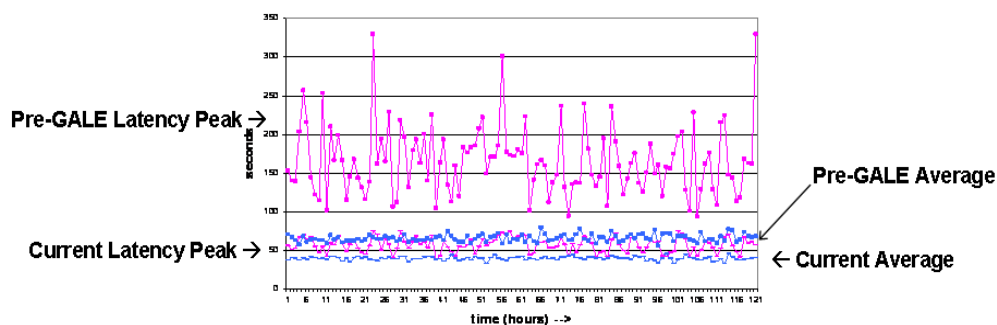


Figure 6.22: Improvement in end-to-end system latency

Figure 6.22 shows the improvements in both peak and average latency (over one hour sessions) due to the new SBD system with explicit length constraints. Peak latency dropped from over 300 seconds to below 80 seconds. The plot also shows the reduction

in latency variation in the discriminative SBD-based AMC engine, which leads to robust operation in the field.

We believe that the improvement in system latency is due to the following reasons:

- An upper limit on the length of the hypothesized sentences,
- An explicit sentence-length constraint in the SBD algorithm results in well-behaved sentence length on average, and
- Incremental Bayesian decoding that does not require completed speaker turns before hypothesizing sentence boundaries.

6.2.3.5. Conclusion

We have demonstrated that the constrained, discriminative SBD system outperforms the previous system, on Arabic and Chinese speech, in terms of sentence segmentation performance. Adding sentence-length constraints has minimal effect on the sentence segmentation performance. However, the peak system latency during live audio processing is significantly reduced.

Perceptron models have a simple formulation, are easy to train, and perform surprisingly well. The Perceptron model we used for SBD was impoverished since we did not use all the available prosodic features. It would be interesting to learn if adding more prosodic features helps the Perceptron model. Model combination has always been beneficial for SBD. We need to examine more discriminative models that might combine effectively with the Perceptron model. Finally, we used a very simple logistic regression model to compute the sentence-length-based penalty score. Further research could examine a machine learning approach to compute the parameters of a similar model or propose new models for imposing length-based constraints in SBD systems.

The Sentence Boundary Detection research described in this section was implemented in the AMC engine, which is offered as a turnkey product by BBN. The length of sentences in the new system and the quality of the segmentation has far-reaching implications for solutions that integrate operational engines in GALE or beyond. These improvements have been partially responsible for the operational viability and robustness of the BBN operational engine, the AMC. Sentences hypothesized in Arabic and Chinese speech by this new discriminative SBD system were used as the common segmentation across the components for the GALE Interoperability Demo (IOD) (Pitrelli *et al.* 2008a). The AMC is also the core Rich Transcription engine for the BBN Broadcast Monitoring System, which is a turnkey product developed in partnership with the Technical Support Working Group (TSWG), which allows intelligence analysts to capture, monitor, and triage information from foreign television stations in real-time with low latency. The BBN BMS is deployed at multiple U.S. Government installations around the world and has been instrumental in transitioning GALE technology and research improvements into the field to aid in counter-terrorism, intelligence analysis, and language learning.

6.2.4. System Combination of Distributed Speech-to-Text, Translation, and Information Extraction Engines: The GALE Interoperability Demo (IOD)

Authors: John F. Pitrelli, Salim Roukos, Edward A. Epstein, David Ferrucci, Greg Hanneman, Kenneth Heafield, Daniel Kiecza, Alon Lavie, Burn L. Lewis, Amit Srivastava, and Paola Virga

6.2.4.1. Introduction

Speech- and text-processing engines, such as speech-to-text and machine translation, are being assembled into ever-larger groups of engines, to achieve several goals:

- to perform increasingly-complex tasks, such as transcription, translation into English, and extraction of information from foreign-language news broadcasts,
- to facilitate system-combination techniques (Jayaraman *et al.* 2005; Fiscus 1997) to achieve higher accuracy, and
- to provide fault-tolerance in case of failure of an engine.

GALE's Language Exploitation Environment (LEE) working group has embarked on several activities to achieve interoperation of GALE engines using the Unstructured Information Management Architecture (UIMA) software framework (Ferrucci and Lally 2004). By "interoperation", we refer to single-point invocation of a group of engines, which we refer to as an "aggregate". The GALE Type System (GTS) (Pitrelli *et al.* 2008b) has been developed as a common set of data formats for such UIMA aggregates. The GALE Interoperability Demo (IOD) system (Pitrelli *et al.* 2008a) employs UIMA and GTS to interoperate 15 engines from seven remote sites spanning all three GALE consortia, to make Arabic broadcast news and Web sites browsable as English text grouped and summarized by topic, and playable as synthesized English speech. IOD inter-operates engines performing 11 distinct functions:

- language/dialect identification,
- gender determination / speaker detection,
- speech-to-text (STT),
- entity detection (ED),
- machine translation (MT),
- translation system combination (multi-engine machine translation or MEMT),
- story-boundary detection,
- topic-clustering of stories,
- summarization of topic-clusters of stories,
- headline generation, and
- text-to-speech synthesis.

These engines run on diverse computing platforms physically distributed across multiple states and countries at their home sites:

- IBM, New York,
- Carnegie Mellon University, Pennsylvania,
- BBN, Massachusetts,
- RWTH Aachen University, Germany,
- University of Massachusetts at Amherst,
- Columbia University, New York, and
- Systran, France.

Several principles guide the design of IOD, as more-fully detailed previously (Pitrelli *et al.* 2008a):

1. Fault-tolerance: When practical, multiple engines performing the same function are employed, to yield robustness against failure of individual engines.
2. Complementarity of engines: Different engines performing the same function may each in turn be best suited to process different types of input data, and so interoperating them followed by a system-combination engine can yield improved accuracy.
3. Exploitation of multiple engine functions: An engine should take advantage of functions preceding it in the aggregate, by making use of their results when appropriate.
4. Well-delineated functions: Some combinations of engines were tightly coupled and mutually optimized, e.g., STT and MT integrated to process audio. Mutual optimization of engines should be exploited for accuracy when practical, however, not in cases in which such coupling impedes interoperability with other engines, as IOD's goal is to achieve accuracy and functionality through interoperation.

In short, IOD's goal is to use UIMA to interoperate engines, to exploit the complementary performance and function of engines by imposing minimal constraints upon each engine, such as input-output specifications to delineate functionality.

For three years, the IOD aggregate of GALE engines has been invoked nightly, processing the audio from three to six hours of broadcast news from two Arabic news networks, Al Arabiya and Al Jazeera. A simpler “web aggregate” is also being run to process text from Arabic news Web sites. In addition, the UIMA Component Repository (UCR) and UIMA Component Container (UCC), described in Section 6.2.5, have been established at Carnegie Mellon University to provide GALE community members with Web-based access to configure their own aggregates of these and other engines to process audio and text data of their choosing.

6.2.4.2. UIMA: Managing Engine Interoperation

The widely-distributed engines that make up the IOD operate under the management of UIMA (Ferrucci and Lally 2004). UIMA is an architecture and software framework for

composing and deploying multi-modal analytics, adding structure to unstructured data such as text, audio, and video. The architecture is undergoing a standardization effort, referred to as the UIMA specification, by a technical committee within the Organization for the Advancement of Structured Information Standards (OASIS).³ IOD uses Apache UIMA, an open-source implementation of the UIMA Architecture.⁴

Capabilities

Some of the features Apache UIMA provides are:

- An extensible type-system mechanism used to formalize the format for all input and output data,
- A common analysis structure to organize and maintain a segment of data, (e.g., text, audio) and all the analysis results,
- An extensible component-based framework that simplifies the integration and deployment of UIMA-compliant analytics,
- Ability to create UIMA-compliant components from analytics written in Java, C++, Perl, Python and Tcl,
- Support for Linux, Windows and other Unix-based OS,
- Tools for developing and testing components individually,
- Ability to run components as shared services on the Internet,
- Ability to create custom analysis flow, with complex error-handling options,
- Ability to process multiple data segments simultaneously through different stages in the aggregate, for increased throughput,
- Ability to process the same segment through multiple engines in parallel to reduce latency, and
- Ability to re-segment data.

UIMA's capabilities are employed by IOD as follows.

Data and Views

Data is passed among UIMA components in the Common Analysis Structure (CAS) with the format and structure of the data defined by an application-specific type system. The GALE Type System (GTS) (Pitrelli *et al.* 2008b) has been created for aggregates of speech- and natural-language-processing engines, such as IOD. It is important to note that agreeing on a type system does not constitute a complete data-flow design for an aggregate such as IOD. Rather, it must be combined with usage conventions, such as, which types or attributes of types are optional and which required for each engine.

Each CAS contains a “view” of the data being analyzed – for the IOD, a two-minute segment of audio from a news video, or a Web page. For multi-modal applications, UIMA supports multiple views within a CAS, each representing a different way to view the same basic data. In the IOD we create additional views as the CAS progresses

³ UIMA Architecture at OASIS: http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=uima

⁴ Apache UIMA: <http://incubator.apache.org/uima>

through the aggregate – a source-language text view for the Arabic text from each STT engine, and a single target-language view for the English text comprising the merged translation. Each view initially contains only the data to be analyzed; engines add their results as annotations on the data.

Most engines need access to only one type of view, and are not dependent on the existence of other views. STT uses just the audio view, MT and MEMT the source-language views, and some monolingual engines such as summarization use the target-language view.

Data Re-segmentation Capability

UIMA supports a powerful type of component, the CAS Multiplier, which has the ability to produce new CASes. For each input CAS processed, a CAS Multiplier can emit zero or more new CASes, effectively splitting, merging, delaying, or arbitrarily re-segmenting the input CASes. If an analytic needs more contextual information than is available in a single CAS, it can be implemented as a CAS Multiplier, releasing delayed CASes when sufficient context has been received. IOD initially segments audio data into manageable but arbitrary segments, and later uses a CAS Multiplier to re-segment the data into complete stories for the convenience of the analytics that follow.

Scale-out, Load Balancing, and Parallelism

Apache UIMA provides a flexible and powerful scale-out capability enabling UIMA components to be deployed as shared services, running in separate processes on the same or different machines. Apache UIMA uses the Apache ActiveMQ implementation of Java Message Service (JMS) to provide asynchronous connectivity between an application and its services. Multiple instances of a UIMA service can be deployed on the same or different machines, all processing CASes from the same input queue, hence providing dynamic load balancing. All components, the dedicated local ones as well as the shared remote ones, have their own input queue and processing thread, so multiple CASes can be active simultaneously, increasing throughput. Since STT is by far the slowest engine function in IOD, this scale-out capability has been exploited by deploying up to five instances of the slowest engines in order to achieve throughput close to real-time.

If an application has a set of engines with no processing interdependences, UIMA can invoke them in parallel, reducing latency. The same CAS is sent to all, and UIMA merges the results returned from each. The engines must only add new entries to the CAS, leaving all existing data unmodified. In the IOD this is done for all of the STT engines, reducing their contribution to the latency to that of the slowest.

Data-reorganization Components

IOD requires data-reorganization components for a variety of purposes, including:

- the creation of CASes at the beginning of the aggregate, and extraction of information from CASes at the end,
- creating, from the early engines' annotations, new views suitable for processing by subsequent engines, along with cross-references back to the original annotations,

- creating data structures within views to prepare for processing by subsequent engines, and
- re-segmenting data into new CASes as needed by subsequent engines.

From a research perspective, these components are very distinct from the engines in the high-level design of IOD. They perform relatively “mechanical” functions interfacing among the engines, as well as between the engines and the outside application. In contrast, “engines” are being improved experimentally on an ongoing basis in distributed research labs, and so are most conveniently hosted at their home sites. Despite this distinction between engines and data-reorganization components, from the perspective of the UIMA framework, all of these are simply “UIMA components”, simplifying development, debugging, and deployment. Specific details of how engines and data-reorganization components interact in the IOD aggregate are described in order of process flow in the following section.

Flow Control and Error Handling

UIMA supports a customizable flow controller that manages the progress of each CAS through the sequence of components. It can specify which components may be run in parallel, and can change the flow depending on CAS content or on engine failure. UIMA supports a number of error-handling operations for each CAS sent to a remote service, e.g. a limit on processing time, the number of retry attempts, the number of errors that will trigger removal of the component or termination of the application, and also if a CAS can continue in the flow after the failure. If the failure of an engine does not affect any succeeding engines or the overall application, or if there are other engines performing the same function, then error handling should let the application continue processing that CAS. In this way, multiple engines provide fault tolerance. In the extreme case when all engines in the set fail, and the function is critical, the flow controller terminates the application.

For the configuration depicted on the left side of Figure 6.23, a pre-processing step extracts two-minute audio segments from Arabic broadcast news video. The news show is later re-segmented into story segments as described below. The configuration on the right side of the figure uses many of the same components, along with an HTML detagger, to process Arabic news Web pages output by a Web crawler. Engines are operated as remote UIMA services, deployed, maintained and upgraded by their home sites in order to keep current with latest research developments. IOD users operate as UIMA clients in accessing these engines. One user is IBM, which runs the audio aggregate every night on current news sources and the Web aggregate on demand. Another is the UCC (Section 6.2.5) at Carnegie Mellon University, which allows users to configure their own aggregates from the IOD engine services, and to supply them with audio or text files.

The following paragraphs detail the operation of the engines and the data-reorganization components interleaved among them to facilitate their interoperation.

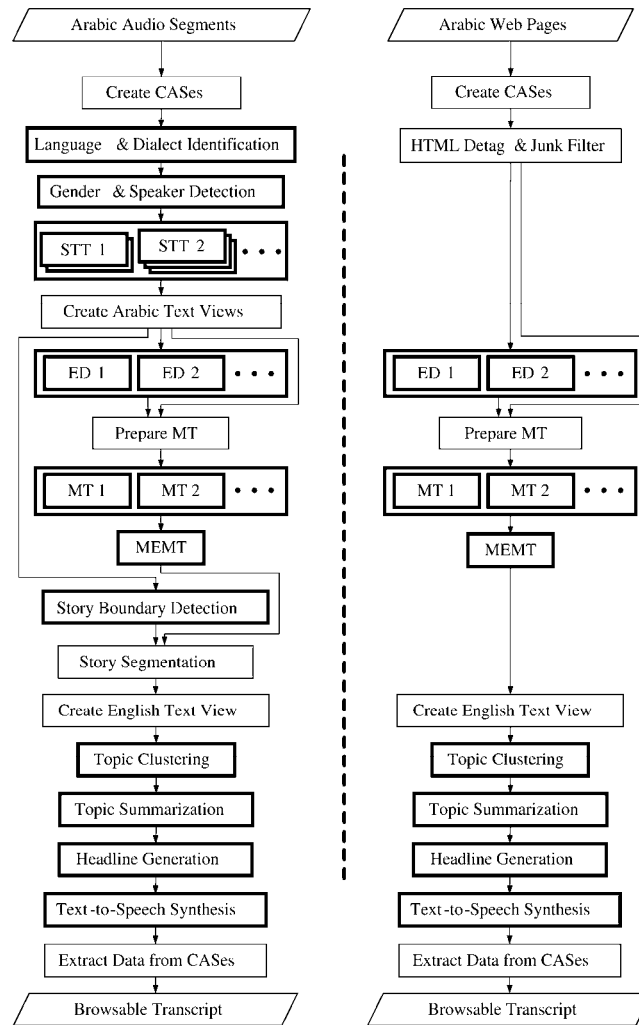


Figure 6.23: IOD system diagram. Bold boxes indicate engines; other rectangles represent data-reorganization components. Arrows represent data dependencies; processing flow is top-to-bottom.

CAS-creation data-reorganization components (UIMA “collection readers”) populate each CAS with the data to be analyzed. The “audio collection reader” creates a CAS for each two-minute audio segment, initializing it with an audio view and an `AudioDocID` containing attributes which represent data about the entire segment, such as that this is the N th out of M segments in a show.⁵ The “web collection reader” creates a CAS for each news Web page, and a following HTML detagger strips out HTML tags and other

⁵ Typewriter font is used for names of data types defined by the GALE Type System; for more details on these types, please consult the GTS paper (Pitrelli *et al.* 2008b).

material to create a source-language text view, representing the segment as an Arabic plain-text string.

The *Language/dialect identification* engine creates `AudioLanguageID` objects in the audio view, identifying the language and dialect between two time points in the audio.

The *Gender/speaker detection* engine creates `SpeakerID` objects in the audio view, each labeling the span between two time points as containing speech by a specified gender, or by a speaker represented in the engine's models.

Speech-to-text (STT) engines add objects to the audio view as follows:

`AudioTokens`, which label a word of speech onto a span between two time points (mandatory)

`SUs` (sentence units), which label a span of time as containing one sentence of speech (preferred),

`SpeakerIDs` (optional), and

`AudioLanguageIDs` (optional).

Note that UIMA aggregation may be hierarchical: what an aggregate considers an “engine” can in fact be another aggregate. An example is that the IOD aggregate's STT “engines” producing the above types are typically implemented as an aggregate, for example, of a speech-recognition engine producing `AudioTokens` and a sentence-boundary-detection engine producing `SUs`.

STT engines are normally the longest running components in IOD. Therefore, they are a priority for deploying multiple instances of an engine, as depicted by stacked boxes in Figure 6.23, currently ranging up to five instances per engine.

Create source-language views is a data-reorganization component creating an Arabic text view corresponding to each STT engine. For each STT, the strings in its `AudioTokens` are concatenated into a single Arabic text string, which serves as a new representation of the content of the segment, to prepare for text only engines such as ED and MT. This component then creates in each new view a set of `AudioXrefs`, each associating a span of characters in the new view with a span of time in the audio view. It also creates `Sentence` annotations, which indicate that a span of string indices in the view corresponds to a sentence. These are derived by taking the `SU` boundary times from a selected STT engine and mapping them across the view alignment onto positions in the new view's text string.

Entity-Detection (ED) engines process each source-language view and create `EntityMention` objects associating an entity type such as “person” with a span of characters in the view. Optionally, ED engines may produce `Entity` objects reflecting coreference analysis associating a set of mentions with a single underlying entity. Also optionally, ED engines may produce `EventMentions` and `Events` analogous to the corresponding entity types, and `RelationMention` and `Relation` objects relating pairs of entities. In Web aggregates, an ED engine is responsible for creating `Sentence` objects.

Prepare MT is a data-reorganization component which creates objects needed by MT engines, specifically, `WordToken` annotations which identify that a span of characters in the view string is a word, and `Translatable` annotations which serve to specify a

partitioning of text in each source-language view into units to be processed by MT engines. Currently, one `Translatable` is created for each `Sentence`.

Machine Translation (MT) engines place target-language text strings in the CAS within one `TranslationResult` for each `Translatable` in each source-language view. In addition, MT engines create `Alignment` structures linking target-language words back to source-language words.

The *Multi-Engine Machine Translation (MEMT)* engine creates a single translation of the content by mining the $N \times M$ translations resulting from the N STT engines and M MT engines. MEMT formats its output translation as `TranslationResults` and `Alignments` just as MT engines do. As MEMT's function is inherently related to multiple-engine processing, it is therefore particularly relevant to interoperation, and so MEMT is described in further detail below.

The *Story-Boundary Detection (SBD)* engine annotates each sentence boundary in a source language view with a score indicating how likely it is to be a story boundary. It requires two minutes of audio context before and after each boundary, so is implemented as a CAS Multiplier in order to delay the processing of each segment until the next one has been received.

Story Segmentation is a data-reorganization component which takes in two-minute segments and emits story segments, and is therefore a CAS Multiplier. Boundaries are chosen by applying a threshold to the scores produced by SBD. Re-segmentation enables subsequent engines, such as topic clustering, to operate directly on the appropriate data, i.e. a complete story. Note: Re-segmentation is not necessary in Web applications since each Web page is assumed to comprise a story.

Create target-language view is a data-reorganization component creating an English text view in each video and Web segment by concatenating the `TranslationResult` strings corresponding to a selected translation, typically that output by MEMT. This view enables monolingual-text functions like summarization and headline generation to focus on English text as their natural input. It also creates entity-related objects in the new view corresponding to any it finds in the source-language view, and creates cross-reference objects linking the new view and the appropriate source-language view.

The *Topic Clustering* engine creates a single `Topic` object in each video-story segment and Web segment. `Topic` contains an identifier which represents a cluster of stories it determines to be about the same topic. For each client, the topic-clustering engine maintains a history about past segments in order to label the current segment.

The *Summarization* engine also maintains a history and gives each CAS a `Summary` object containing a string providing an English summary of all stories in history to date sharing topic identifiers with the current CAS.

The *Headline Generation* engine creates two objects in a typical CAS: a `SummaryHeadline` containing a headline derived from the CAS's topic summary, and a `StoryHeadline` derived from the story in the CAS.

The *Text-to-Speech Synthesis* engine creates an audio rendition of the target-language view string, matching detected gender in the case of a video segment. This engine makes use of multiple views, as it follows cross-reference links between the views to associate each target-language sentence with the gender annotations applied to the corresponding audio span.

Extracting Data from CASes are UIMA CAS consumers which end the aggregates by extracting the results from each CAS, saving the information in an XML structure enabling users of a custom Web application to browse the topics, stories, and parallel translations.

Sliding Results Window

Currently, IOD is run every night and maintains a four-day results window. The intermediate results prior to the first history-maintaining engine, topic clustering, are saved for three days, so that four days worth of data can be run through the final stages to produce the daily results.

6.2.4.3. Multi-Engine Machine Translation

Of particular note among IOD's engines is the translation-combination engine, Carnegie Mellon University's multi-engine machine translation (MEMT), because it particularly exploits interoperation of multiple engines. Specifically, IOD typically runs two or three STT and two or three MT engines; each resulting STT–MT system combination provides another possible target-language translation for any given segment of source-language audio. MEMT collects the translations resulting from each STT–MT combination in the IOD aggregate, and assembles a new synthetic translation that tends to be better than that from any single STT–MT output. To carry out fast online synthetic combination within the aggregate, it relies on minimal information from the incoming translations, requiring only the text string produced by each STT–MT combination and (optionally) alignment information tracing each target-language word back to the source-language word that produced it.

Combination Approach

The translations for each source sentence are first word-aligned to each other based on exact string match, stem match, or synonymy match using WordNet synsets. MEMT then iteratively builds up collections of partial synthetic hypotheses of increasing length by selecting an unused word from one of the original inputs and adding it to the end of a hypothesis currently under construction. When a hypothesis is completed, it is scored and added to a list of final hypotheses. At the end of the process, the highest scoring hypothesis is selected as the MEMT output and is passed to the remainder of the IOD aggregate. Special constraints can be enforced for the creation of atomic phrases, or high quality contiguous sequences of words in an original input that we would like to keep together as a single unit in the synthetic MEMT output.

Although we have preferred the synthetic combination method from the beginning (Jayaraman *et al.* 2005) as a way to produce a more flexible word ordering in the MEMT output, the MEMT system can be configured to perform hypothesis selection or fixed-word-order combination in the style of a confusion network (Rosti *et al.* 2007).

Integration into the IOD

The online and distributed nature of the IOD environment enforces some key operating constraints on the MEMT system, while the richness of the data defined within

the GALE Type System (GTS) allows the algorithm to take additional information into account when constructing synthetic combinations.

MEMT is designed to be robust to variation in number of translations of a sentence, and can handle cases in which some of the input translation alternatives are empty, due to the possibility of STT/MT engines being lost due to system or network failure.

GTS includes detailed alignment information for mapping from audio segments to recognized source language words to translated targetlanguage words, which is maintained throughout the IOD's processing. This presents a particular challenge for MEMT, as it is simultaneously dealing with translations that were produced using different text views of the same audio, each of which has a different set of alignments. To solve this problem, we augment the output of the MEMT combination to additionally specify, for each word in its synthetic output, which word instance from which MT output was chosen. Our system is then able to assemble a consistent and correct set of alignments for the MEMT output by recovering the original alignment information for those words.

6.2.4.4. Adapting Existing Engines to UIMA Interoperation: Case Study, BBN STT

BBN's Audio Monitoring Component (AMC) product produces in real-time a continuous rich transcript of foreign-language broadcasts.⁶ The transcript includes speaker information, sentence boundaries, and named entities. The integration of AMC into the IOD/UIMA framework was completed rapidly for several reasons: (1) AMC is a mature product producing XML-formatted transcripts which adhere to a strict schema contract, (2) GTS required only a subset of the AMC schema types and mapping these types was straightforward, and (3) UIMA provided the necessary infrastructure for quick integration, including simple framework installation, an API to generate annotations, and scripting language support.

The MEMT component in IOD requires common sentence segmentation for all STT and MT engines. The sentence segmentation produced by AMC was adopted as a common segmentation for the demo for several reasons: The AMC sentence segmentation is produced by a tightly-coupled, rich transcription engine. Also, and more importantly, the AMC sentence segmentation is specifically tuned for human readability. Its sentences will, for example, not exceed a fixed upper-duration threshold. This behavior proved critical to ensure reliable performance of the MEMT component. AMC for IOD is hosted at BBN Technologies. Incorporation of AMC's ED remains work in progress, due to IOD's categorization of ED as a text-annotating rather than a speech-annotating function. However, all other AMC functionality has been part of the entire history of IOD.

⁶ BBN BMS system: http://bbn.com/products_and_services/bbn_broadcast_monitoring_system/ (the BMS includes BBN's AMC STT engine)

6.2.4.5. Experiences and Observations

Constraints on Engines

Running an engine in the context of an aggregate imposes constraints which do not apply to running in isolation. Adding a new engine to an aggregate often creates new constraints on several other engines. Many of these constraints are low-level in nature; discussion here will be limited to illustrative examples.

One example is that an engine's standard output often includes special codes for exceptional situations, such as one MT's "unk" for an unknown word. Isolated evaluations of engines account explicitly for such codes. However, in a growing aggregate, requiring engines to handle others' codes becomes impractical. In this example, subsequent engines may mistreat all appearances of "unk" as multiple occurrences of a single popular word, degrading the accuracy of algorithms dependent on word frequency, so MT needs another scheme for handling unknown words, such as attempting a transliteration.

Another example is that interoperation can magnify the consequences of a malfunctioning engine. Outputting malformed data can cause a compounding of malfunction from the downstream engines. Therefore, interoperation imposes heightened requirements for engines to produce valid data.

Performance Evaluation

Rigorous evaluation of IOD accuracy is impeded by

- the large number of interacting engines,
- the complexity in obtaining ground-truth data against which to assess aggregates' outputs, and
- the presence of several engine functions whose results' accuracy remains subjective, such as some of the text-processing engines.

We have begun to define a formal framework for evaluation of such aggregates (Murthy *et al.* 2008).

Upgrading Engines

Frequently, an engine somewhere in the aggregate is improved, and the improvement is validated by a unit test; however, care must be taken in the presence of a downstream engine whose classifier is trained on output data from the engine to be upgraded, often necessitating ripple-forward re-training.

6.2.4.6. Conclusions

The IOD has demonstrated that it is possible to build a useful application from the combination of many powerful analytic engines deployed on a variety of platforms and languages at sites around the world. The UIMA development environment facilitated the independent development and testing of each component, making it easy for this diverse mixture of companies and universities to collaborate efficiently.

Natural language processing technologies have advanced to the point that large aggregates of NLP engines can provide useful output on tasks requiring such combinations of engines. In spite of their computing-platform variations and distributed locations, UIMA enabled the IOD team of seven GALE sites to collaborate on interoperating 15 engines to transcribe, translate, and extract information from Arabic news video and Web content daily. Results are available showing English text headlines hyper-linked to topic summaries, story headlines, and story translations.⁷

The type system and data-reorganization components developed for IOD could be used as an experimental test bed for a collaborative effort evaluating the effectiveness of various combinations of engines developed by separate groups. Many research areas involving the interoperation of engines could be explored, e.g. the effectiveness of system-combination techniques on transcriptions and entities as well as on translations, the benefits to MT of using the source-language entities, the value of independent gender and/or speaker detection to STT, etc.

Acknowledgements

The authors thank James Allan, Bob Armstrong, Shilpa Arora, Manuel Bardea, Oliver Bender, Eric Brown, Jaroslaw Cwiklik, Bret Ehlert, Gopi Flaherty, Martin Franz, Andre Gauthier, Isaac Harris, Abe Ittycheriah, Nanda Kambhatla, Francis Kubala, Daben Liu, Evgeny Matusov, Kathy McKeown, Justin Merrill, Uma Murthy, Udhyakumar Nallasamy, Eric Nyberg, Leiming Qian, Jerome Quinn, Ganesh Ramaswamy, Eric Riebling, George Saon, Barry Schiffman, Jean Senellart, Sergey Sigelman, Olivier Siohan, Sara Rosenthal, David Svoboda, Alan Watkins, William Wong, and Jian-Ming Xu for their contributions to IOD.

6.2.5. The UIMA Component Repository (UCR) and UIMA Component Container (UCC): A Facility for Flexible Exchange and End-User Configuration of Operational NLP Engines

Authors: Eric Nyberg, Eric Riebling, Burn L. Lewis, and David Svoboda

6.2.5.1. Introduction

As part of our work on the GALE Language Exploitation Environment (LEE), the LEE working group specified a common annotation type system for speech- and text processing engines (Pitrelli *et al.* 2008b) using the Unstructured Information Management Architecture framework (UIMA).⁸ This common type system has been used to integrate a variety of different components from different organizations into the GALE Interoperability Demo (IOD), an aggregate analysis engine that supports daily processing of emerging news in foreign-language video (Pitrelli *et al.* 2008a). Although the

⁷ Video: <http://rosetta.watson.ibm.com:8888/iod/>

Web: <http://rosetta.watson.ibm.com:8888/iod/?web=1>

⁸ The latest version of UIMA is currently being developed as an Apache Incubator project: <http://incubator.apache.org/uima>

development of the shared type system and IOD was motivated primarily by the requirements of the GALE program, the shared type system and aggregate component architecture support a degree of modularity and interoperability which in turn imply a significant opportunity for low-cost component and aggregate reuse.

This section describes two additional capabilities developed by the GALE LEE team: the UIMA Component Repository (UCR), a Web-based portal designed to support efficient component dissemination and reuse; and the UIMA Component Container (UCC), a Web-based portal for run-time execution of aggregate analysis engines.

6.2.5.2. UIMA Component Repository (UCR)

The UIMA Component Repository (UCR) consists of a Web-based user interface to a relational database which stores UIMA components (Collection Readers, Annotators, CAS [Common Analysis Structure] Consumers) along with associated metadata about each component. Users may submit their own component in one of two ways. They may publish only the component's descriptor (an XML file which indicates how the component can be invoked as a service), or if they wish to disseminate the binary (and optionally source) code for their component, users can upload it to the UCR in a special .zip format (UIMA Processing Engine ARchive [PEAR] format). There are two types of users: registered and unregistered. All users may browse and download components or descriptors, but only registered users may upload components. The specific use cases supported by the UCR are listed in the following subsection.

Capabilities

The UCR supports the following user actions:

- *Register User.* New users register by selecting a username and password, and providing a valid email address.
- *Join Mailing List.* The site provides an interface for subscription to a LISTSERV mailing list for UCR users.
- *Browse Components.* The site supports a variety of browsing options for displaying components: All Components, Components by Category, Components by Submitter, and Commercial (costs money) vs. Noncommercial (free) Components.
- *Search Components.* The site provides a simple keyword-based search capability that matches the user's keywords with the component titles and descriptions; an advanced search capability allows the user to specify search terms to be matched against particular component metadata fields (name, description, developer name, input and output annotation types).
- *Download Component or Descriptor.* The user may download either the component code (PEAR file) or XML component descriptor for the component being viewed.
- *View Component Type Tree.* When the annotation type system is provided with the Component (a PEAR file is present), the system provides a capability for viewing the annotation type hierarchy within the Web browser.

- *Rate Component.* A simple 1-5 stars rating system is provided to registered users so that they may provide feedback on individual components; the average rating is displayed when a user displays a component.
- *Announce Component.* The user provides the metadata and descriptor for the component, but does not provide the PEAR file (component code).
- *Publish Component.* The user provides the metadata, descriptor and code for the component, encoded in a UIMA PEAR file.

The UCR also provides additional resources for the user, such as site help, a UIMA FAQ, tutorial videos, and community links to additional off-site resources.

Design

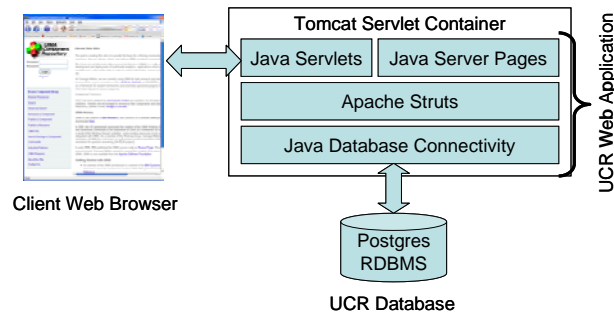


Figure 6.24: UCR Architecture

The UCR is designed as a Web application based on a typical three-tier Model-View-Controller Web application architecture. The underlying data model is a relational database schema which persists information about components and their metadata, users, user roles, type systems, licensing information, and runtime requirements. The primary interface to the UCR is a set of Java servlets, which access static page content via HTML and Java Server Pages, and dynamic content (e.g., results of component searching and browsing) via a set of actions defined in Apache Struts. Struts actions are implemented to use a specialized data access object (DAO), which utilizes the Java Database Connectivity (JDBC) library to access the underlying relational model. The current architecture is shown in Figure 6.24.

Deployment

The UCR was first deployed on the Web in July 2006.⁹ UCR currently runs on Red Hat Linux servers using Apache Tomcat as the servlet container and PostgreSQL as the relational database. In addition to the public deployment, we have also deployed an access-restricted enterprise instance of UCR to support dissemination of components within the GALE community. As of January 2009 the public deployment supports 15

⁹ <http://uima.lti.cs.cmu.edu>

unique component submitters, and 43 distinct components have been uploaded; the broader community includes 87 users who have subscribed to the UCR mailing list.

6.2.5.3. UIMA Component Container

The UIMA Component Container (UCC) was created to support Web configuration and deployment of completed UIMA engine aggregates, which integrate various components of GALE IOD. The UCC allows any user with access to a Web browser and the Internet to upload data and process it with aggregates of IOD engines, eliminating the need to pre-install and manually configure the UIMA SDK and the IOD components, which are accessed as remote Web services. Users may upload speech and/or text data, select components to run, and queue processing jobs which run asynchronously; as jobs are completed, the resulting output is placed in a directory for later retrieval.

Capabilities

The UCC supports the following user actions:

- *Upload Data.* The user may upload data files (text or PCM audio) to be processed.
- *Configure Pipeline.* Based on the input data which has been selected, the user may choose from the available components to be run by the pipeline.
- *Create Job.* The user selects a particular set of input files to be processed by the configured analysis engine.
- *Queue Job.* The user queues the configured job for processing.
- *View Jobs.* The user may view the contents of the job queue, as well as the status of their submitted job(s).
- *Cancel Job.* The user may cancel a running job, which is terminated and removed from the job queue.
- *View Output.* The user may view the text and/or speech output from any completed job.
- *View Log.* The user may view the log for completed as well as in-progress jobs.

Design

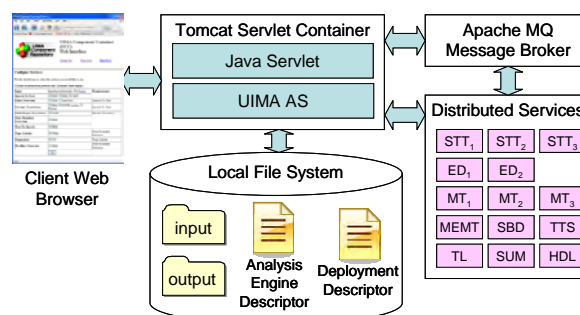


Figure 6.25: UCC Architecture

The UCC is designed as a servlet Web application that stores job input and output on the local file system (see Figure 6.25). Input may be Arabic text or Arabic audio (no other languages), and output is English text and optionally synthetic speech audio. Based on the options selected by the user, an analysis engine descriptor and deployment descriptor are created for the configured pipeline. Configured jobs are queued and executed within an UIMA Asynchronous Engine (AE). The AE runs a local Collection Reader and CAS Consumer to read and write data from/to the local input and output directories; the components in the pipeline are accessed as distributed services. An Apache ActiveMQ message broker instance provides a mapping from the component namespace used in the deployment descriptor to the actual IP addresses of the nodes running the deployed services. This lightweight design does not require that any component code be delivered to the UCC or run locally, simplifying the integration process and reducing the level of processing resources required to deploy the UCC.

The UCC Web interface uses a simple string protocol to transmit the names of selected services to the UCC Web application. Services are named based on provider and service type, and pipelines are specified by appending these name strings together, e.g.:

```
IOD.artxt.IBM-ED.BAS-ED.IBM-MT.RWTH-MT.CMU-MEMT
```

There can be more than one provider for a given service. For example, three different organizations provide Speech-to-Text engines. When more than one provider of the same service is selected, the services are run in parallel. Different services are presented to the user for selection, based on whether the chosen input data is text or audio.

Deployment

The UCC runs as an Apache Tomcat Web application. For each queued job, an analysis engine descriptor (specifying the components) and a deployment descriptor (specifying how to locate the components as Web services and run the pipeline) are created from the string description of the pipeline provided by the HTML forms interface. These descriptors are used to create an instance of a UIMA Asynchronous Engine, which reads the input data and connects to multiple remote Asynchronous Engines that provide various NLP services via the Apache ActiveMQ Message Broker. The current set of services and providers integrated in the UCC IOD pipeline (see Figure 6.25) includes:

- **STT**: Speech-to-Text (BBN, IBM, Carnegie Mellon University)
- **ED**: Entity Detection (IBM, BasisTech)
- **MT**: Machine Translation (IBM, RWTH Aachen, Systran)
- **MEMT**: Multi-Engine Machine Translation (Carnegie Mellon University)
- **SBD**: Story Boundary Detection (IBM)
- **TTS**: Text-to-Speech (IBM)
- **TL**: Topic Labeler (UMA)
- **SUM**: Summarizer (CU)
- **HDL**: Headline Generator (IBM)

6.2.5.4. Experiences and Observations

When components are uploaded to the UCR as PEAR files, they are validated using built-in library code that is part of the UIMA SDK. In practice, some submitted UCR components failed to validate. In addressing this issue, we learned that the validation process is sensitive to the version of Java used to compile the component code, which must be identical to or earlier than the version of Java used to compile and run the UCR. In our case this required upgrading the UCR and the enclosing Tomcat container to run under Java 6.

We also learned that components submitted to the UCC should not depend on external code, services, file structure, or environment variables. We revised the UCC help documentation to encourage developers to test their components using the UIMA PEAR installer before submitting them to the UCR. This mirrors the UCC run-time environment, and has helped developers to create more environmentally independent components.

Incremental revisions to the UCC design were required as more services were added to the IOD pipeline. The original design required developers to run a Perl script to generate Analysis Engine Descriptors and Deployment Descriptors for each possible combination of services, and then run a separate script to remove files for ill-formed service combinations. This process proved unwieldy as the number of possible descriptors grew to several hundred. The revised design allows the UCC to generate custom descriptors based on the services chosen by the user. A file containing Analysis Engine and Deployment Descriptor 'snippets' for each remote AE is used to generate the custom descriptor for each job's specific pipeline. This approach is much more efficient, significantly reducing the number of descriptors that comprise the UCC Web application.

The mechanism for adding new services can still be improved. Although the use of pre-defined descriptor snippets helps, the names of services referenced within the snippets file must be coordinated with service references within the Java code. We plan to refactor the code to use service names which are defined in an easily-maintained central location.

It is difficult to report progress on the processing of individual documents (UIMA CASes) within pipelines configured using the UCC. Because of the presence of CAS Multiplier components (to support parallel execution of services), the number of output CASes can exceed the number of input documents. Although the UCC currently presents "M of N" CASes processed as part of its status display, these numbers do not make sense to the user if M exceeds N. Better approaches may be to simply report the number of CASes currently being processed or the percentage of work completed, based on seconds of audio or characters of text.

Because the Asynchronous Engine instance runs in a separate thread from the UCC servlet (which returns immediately after a job is queued), UCC does not provide dynamic status information during execution; the user must manually refresh a "View Job Status" page. This solution is less than satisfactory for long-running jobs which require constant monitoring (e.g., during debugging). A better solution would be to implement some form of "push" technology (e.g., AJAX) to improve usability.

Due to the nature of the GALE Interoperability Demo, some services (such as the Topic Labeler and Summarizer) are required to save state as streams of documents are processed. Since each remote service receives documents to process as separate service requests, an additional mechanism is required to indicate that a job is a continuation of an earlier job from the same user, so that components know whether to re-initialize any state variables. To address this requirement, we added a new capability to the UCC that allows the user to specify whether a job represents a new session, or whether it is a continuation of a previous session from that user. Another component (Story Boundary Detection) requires that all documents in a job are to be processed in order, and therefore, saves state between documents, as well as, any out-of-order documents. To help SBD recognize overlapping or incomplete jobs UCC adds a unique job ID to each request.

6.2.5.5. Conclusions

We have described the UIMA Component Repository and UIMA Component Container, two capabilities which are designed to promote re-use and easy re-deployment of text and audio analysis components configured as UIMA asynchronous analysis engines. Although the current implementations of UCR and UCC meet the basic capability requirements, there are several areas which are targeted for ongoing improvement:

- The UCR does not explicitly manage or reason about the Java version level and UIMA SDK compliance of submitted components; currently developers must re-compile and re-submit their own components, and users who download components may be required to recompile them.
- The UCR cannot execute a single component on sample test data in order to determine its suitability for an intended task before it is downloaded.
- The UCC requires significant effort to deploy a new pipeline; it would be useful to provide a wizard capability for generation of snippets files for automatic compilation of descriptors.
- The UCC persistence mechanism (using the local file system of the server) will be difficult to scale unless it is re-implemented using a more flexible and powerful persistence mechanism (e.g., a relational database).
- Some users of the UCC may prefer a deployment which supports streaming data to/from the UCC server in real time, eliminating the requirement for manual data upload/selection and local data storage.

We plan to address these issues in ongoing refinements to the UCR and UCC packages.

Acknowledgements

The authors thank Shilpa Arora, Eric Brown, Luis Chavez, Yurdaer Dogonata, David Ferrucci, Tong-Haing Fin, Lei Fong, Greg Hanneman, Kimberly Kettner, Nanda Khambatta, Alon Lavie, Humberto Lezama, Justin Merrill, James Rankin, Hideki Shima, and Django Wexler for their contributions to the UCR and/or UCC.

6.3 Evaluation of Operational Engines

6.3.1. The EDIE Framework for User-centered Evaluation of GALE Systems

Authors: Daqing He, Peter Brusilovsky, Jonathan Grady, Jaewook Ahn, Yiming Yang, and Monica Rogati

6.3.1.1. Introduction

As part of our participation in the development of evaluation tools for GALE systems, we developed the “Exploration and Discovery Information Evaluation” (EDIE) framework to assist with the development of the Rosetta system and its components.¹⁰ Along with the series of experiments testing the Rosetta system, we also developed associated evaluation methodologies and metrics, which make EDIE a complete framework for user-centered utility evaluation.

The GALE program aims to develop and apply technologies to deliver pertinent and consolidated multilingual/multimedia information in easy-to-understand forms for military personnel and monolingual English-speaking analysts to use in response to direct or implicit requests. Therefore, utility evaluation, which specifically examines how well the technologies can facilitate such delivery, is vital to the development of GALE systems. Our EDIE work concentrated on informing the design and development of GALE systems through user-centered utility evaluation. Our primary application focused on the Rosetta system and the Carnegie Mellon Adaptive Filtering Engine (CAFÉ). Our experience shows that EDIE itself is a valuable contribution to the GALE program. The goal of this section, therefore, is to introduce EDIE and the accompanying evaluation design principles to the GALE community.

6.3.1.2. EDIE for Evaluating GALE Systems

EDIE was designed at the beginning of the GALE program to enhance the development of the GALE systems by providing robust utility evaluation. At that point, the utility evaluation issues in the GALE systems were still at the discussion stage. The fundamental challenges that EDIE, as an evaluation framework, was designed to address are: can any existing information access tasks be borrowed to model GALE user context? Can a set of task characteristics be proposed to simulate a least one type of GALE tasks? How should a performance-based user evaluation of GALE systems be organized?

6.3.1.2.1. GALE Analyst’s Activity as Task-based Information Exploration

When interacting with a GALE engine, the information needs and corresponding search processes of military personnel and intelligence analysts are often heavily influenced by the complex tasks assigned. These types of search activities can be modeled as task-based information exploration (TBIE), which is a primary research topic

¹⁰ See section 6.3.2.2 for details on the design of the Rosetta system.

in information access (White *et al.* 2006). Existing evaluation frameworks such as Text REtrieval Conference (TREC) are limited in evaluating TBIE, because the metrics and the methodologies are not adequate in the context of highly interactive exploratory search.

6.3.1.2.2. The Simulated Tasks

In order to achieve realistic user-centered utility evaluation of GALE systems, the EDIE framework simulates the tasks that an analyst performs while working with this class of systems. The following characteristics were assumed:

- *EDIE tasks resemble real-world work assigned to analysts.* All EDIE tasks start with a topic statement that simulates a Request For Information (RFI), and result in a short summary that resembles a two page “point paper.” A subject has to explore large volumes of data from various sources to generate such a summary.
- *EDIE tasks are complex.* Each task requests information about a seminal event and its associated aspects. Therefore, these tasks usually include several subtasks that are connected to the overall task.
- *EDIE tasks are dynamic.* Events naturally evolve over time, so an EDIE task has to track the development of the events over several sessions before the final point paper can be generated.
- *EDIE tasks collect information at sub-document (snippet) level.* Although documents still play important roles in GALE searches, the useful information is typically contained in sentences or paragraphs (called snippets). With their small size, but high density of useful information, these snippets are perfect for investigation or report writing.

Among the existing test collections, we chose the TDT4 corpus developed for the Topic Detection and Tracking evaluation (Allan 2002) as the initial dataset in EDIE. We redesigned 18 EDIE tasks based on TDT topics using the assumed characteristics noted above as the guidelines.

In addition to the description of the task and subtasks, each EDIE task also contains brief background information, including a seed story (i.e., a document relevant to the scenario). This helps to establish some consistency in the minimum background knowledge. Figure 6.26 shows a real EDIE task.

G40001 – Galapagos Oil Spill

Background:
San Cristobal is a place near the Galapagos Islands in the Pacific Ocean.

Short description of the task:
An Ecuadorian oil tanker has spilled fuel near the Galapagos Islands, affecting the local fishing industry and threatening local beaches and wildlife. Your tasks are to suggest 1) how and where ocean currents may spread the oil, and 2) what actions should to be taken after the oil leak event caused by the oil tanker Jessica near the Galapagos Island. You should investigate possible support that can be provided by the U.S., as well as potential legal actions against the responsible persons.

From the documents, find snippets of text that contain answers to each of the following questions:

1. On what date did the oil spill occur?
2. Where is the location of the oil spill?
3. What type of vehicle was involved in the event?
4. How many gallons of fuel were spilled?
5. What is the cause of the spill?
6. Who is responsible for the spill?
7. What animals have been affected by the leaked fuel?
8. Who put efforts in relieving the pollution?
9. Any support from other countries?
10. What is the impact on the local people, environment, animals and plant life?
11. What are the penalties/lawsuits for those responsible?

Figure 6.26: An EDIE task

Human annotators were recruited to mark up the “ground truth” for each EDIE task (i.e., to mark-up the documents relevant to each task). To reduce subjectivity, at least two annotators worked on each task.

The annotations were collected based on using a much higher level of detail than a traditional document-level ground truth:

- annotations were made at the snippet level;
- annotations were independently collected under three aspects: topical relevance, novelty, and utility. We ask the annotator to provide utility annotation directly to us so that we may study the utility as a function of combining topic relevance and novelty.
- annotations were collected at three levels of judgment: highly relevant/novel/useful, slightly relevant/novel/useful, and not relevant/novel/useful;

In total, we annotated snippets for all 18 tasks. The annotation files are available at <http://amber.exp.sis.pitt.edu/gale/GALE-resources.html>

6.3.1.2.3. Human-Centered, Multi-Stage, Multi-Aspect Utility Evaluation

EDIE assumes controlled laboratory experiments to assess the GALE systems. As the GALE systems are highly interactive, evaluations using EDIE do require that human subjects conduct task-based exploration and interact with the information system. Human subjects' interaction behaviors and interaction processes are important aspects for observation and evaluation.

More importantly, EDIE assumes that analysts' information exploration behaviors contain two major stages (or loops): *information foraging* and *sense-making* (Pirolli and Card 2005). During the foraging stage, analysts use their domain and search knowledge to collect potentially useful information from various media and sources. In the sense-making stage, analysts then process, organize, and distill the collected information into a coherent form so that it can be integrated into their state of knowledge. Because of the different nature of foraging and sense-making, we believe that the evaluation of systems, which targets information analysis, should consider both stages.

As indicated by the annotation work presented in the section about simulated tasks, EDIE focuses on examining multiple aspects of the utility of the selected snippets. Both the topical relevance and the novelty of the snippets are critical. In addition, we examine the final utility of the snippets by tracking whether and how the snippets are used by the analysts in their final reports. To capture the event development and the need to handle duplicate information, EDIE also simulates the multiple sessions in utility search.

6.3.1.2.4. Utility Evaluation Metrics

EDIE adopted several evaluation measures. The first type of measures focused on *system retrieval effectiveness*. This includes measuring the accuracy and coverage of identified information (i.e., precision and recall) and measuring the utility of the obtained information for a specific task. All these measures concentrate on snippet level examination, rather than document level.

An example of the proposed measures of system performance is a snippet level precision that takes advantage of the fact that two human annotators generated the ground truth. The formula (6.10), which is derived from the work of Allan (2003), calculates the precision of a snippet i selected by a human subject, where oll_i is the character length of the common text chunk between the snippet i and the corresponding ground truth; w_i is the weight of the ground truth combining the mark-ups of both annotators. The value of the weight w_i can be one of five ad hoc assigned levels: 0, 0.25, 0.5, 1, 1.25, 2. nml_i is the character length of the part of the snippet i that has no overlap with the ground truth. Here the 0.5 associated with nml_i is the penalty weight.

$$\frac{\sum_i oll_i \times w_i}{\sum_i oll_i \times w_i + \sum_i nml_i \times 0.5} \quad (6.10)$$

The second type of measures focused on *formal user performance*, on *user activity patterns* mined from transaction logs, and on *subjective evaluation* obtained via questionnaires and interviews. Our idea of formal user performance is similar to that of Marchionini and Shneiderman (1988), where the measures examine the paths and

decisions taken by the users during the process of completing their tasks. All these measures have been expanded to evaluate the information foraging and sense-making stages separately.

The utility measure in EDIE also deserves special mention (Yang *et al.* 2007). To be most precise, utility should be judged on the point paper, which is the final product of the task. However, the varying abilities of the subjects to produce a coherent report are beyond the scope of evaluating a GALE system. Therefore, EDIE's utility measure examines both: the process of snippet selection (i.e., foraging) and organization (i.e., sense-making). Together, the quality of snippet selection and organization reflect the system's utility. The measure also imposes a word limit on the final selected snippets so that the final snippet selections more closely resemble the point paper in size. At the same time, the word limit brings in a cost function for tradeoff - should it include as much available relevant information to a specific question or should it cover as many required questions within the specified word limit?

6.3.1.3. Applications of EDIE in Evaluating GALE Systems

In this section, we will briefly present the involvement of EDIE in evaluating several GALE systems and components. The goal is to reveal EDIE's usages in the context of the GALE systems.

6.3.1.3.1. CAFÉ Utility Evaluation

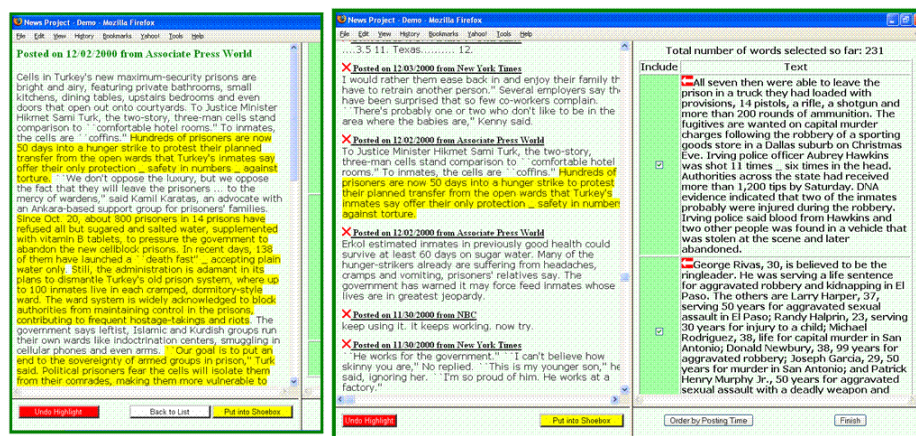


Figure 6.27: Information foraging in CAFÉ system: assembling text fragments in the shoebox

This evaluation examined an innovative adaptive filtering engine CAFÉ, which is one of the major engines supporting Rosetta (Yang *et al.* 2007). Figure 6.27 shows the information foraging interface in CAFÉ. The left side of the right window shows the snippets recommended for examination. Analysts can select any part of the snippets and add them to the right side of the right window. The right side is called "the shoebox," which is a container for holding the snippets that the analysts find potentially useful for the final results. At any time, the analysts can click on a snippet to see the corresponding full document in a pop-up window, which is shown as the left window in Figure 6.27.

Figure 6.28 shows the sense-making interface for CAFÉ. Here, all the snippets in the shoebox that were collected by the analysts during the information foraging stage are displayed at the left side of the screen. The analysts can delete any snippet, or label the snippet as related to one of the sub-task questions. Once the snippet is labeled, it will appear at the right side of the screen which shows the content associated with each sub-task question.

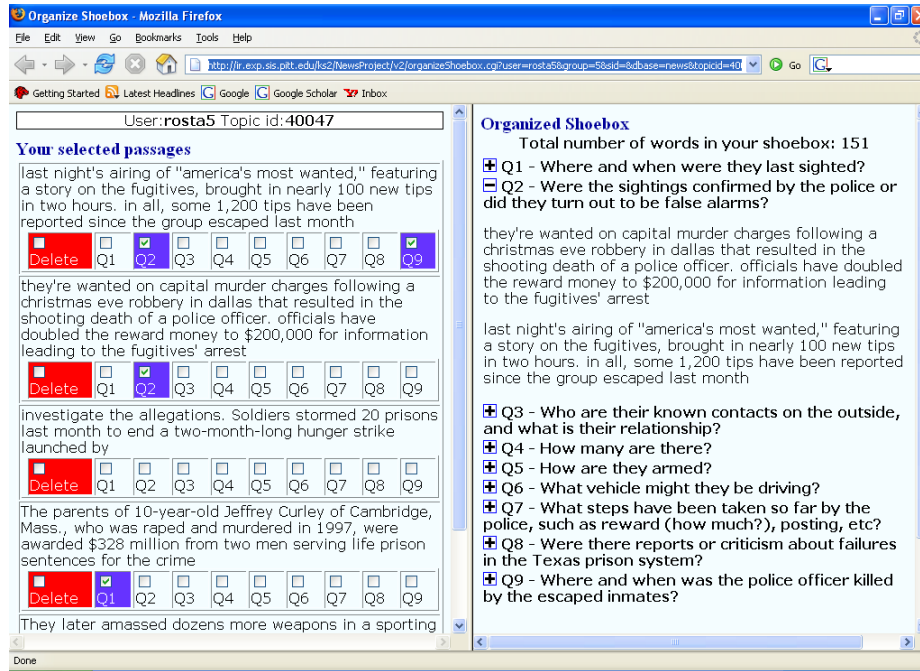


Figure 6.28: Sense-making in CAFÉ: selecting and organizing fragments in the shoebox

The goal of this evaluation was to confirm our hypothesis that adaptive filtering provides value in both stages of GALE searches (at least in comparison with a traditional Google-like search) and to measure the effect of CAFÉ. This study was conducted with eight subjects in July and August, 2006. The baseline system was a state-of-the-art snippet retrieval engine based on Indri (He and Demner-Fushman, 2003). Eight tasks were selected among the 18 possible scenarios.

Our study confirmed the usefulness of the CAFÉ engine, and also confirmed that there is a strong need to have adaptive engines in GALE systems. The results demonstrated that CAFÉ can generate ranked lists of snippets of significantly higher precision than the baseline system. CAFÉ users waste significantly less effort finding useful information and could find more useful information in various stages of their exploration. The details of this CAFÉ study were published by He *et al.* (2008).

6.3.1.3.2. Evaluation of CAFÉ's Profile Update Model

The goal of this study was to assess the value of keeping the list of recommended items up-to-date in the information foraging stage of a GALE search. To make the

appropriate design decision for CAFÉ and similar GALE systems, it was important to know how frequently the user's profile and the list of recommended items should be updated in GALE. The literature analysis revealed two popular approaches: some systems do it once per session (thus, it is called the *between-session* update strategy), whereas, others update whenever there is feedback entered into the system (called the *instant update* strategy).

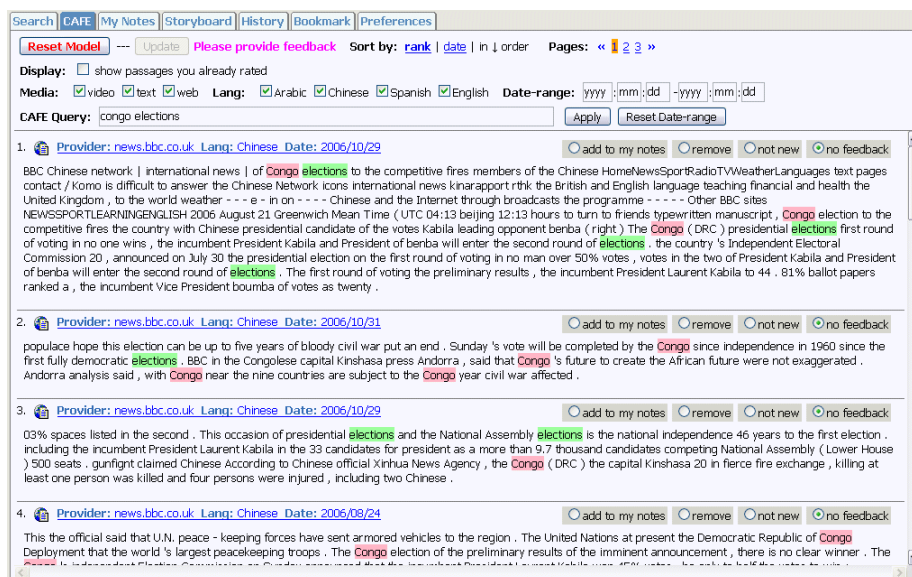


Figure 6.29: CAFÉ engine in Rosetta system

To choose the approach which works best for the GALE system, we designed a controlled experiment with human users performing realistic tasks using two versions of the Rosetta CAFÉ engine that have the same underlying adaptive filtering engine, but with two different update strategies. Figure 6.27 and Figure 6.28 show the Rosetta CAFÉ engine with the between-session update strategy. Figure 6.29 shows the Rosetta CAFÉ engine with the instant update strategy.

The study was conducted in the summer and winter of 2006. We recruited 15 subjects and used two task scenarios from EDIE.

Our results demonstrated that the optimal update strategy to achieve both adaptability and stability for GALE system lies between the instant update and the between-session strategies. The between-session strategy helped to find information with better utility, and received better responses from subjects about its usefulness and usability. However, it prolonged the selection of useful snippets, whereas the instant update strategy helped subjects to obtain almost all of their selected snippets (> 98%) within the first five minutes of the search sessions. The detail of this study was published in by He, Brusilovsky, Grady, Li, and Ahn (2007).

6.3.1.3.3. Evaluation of Task Model Usage

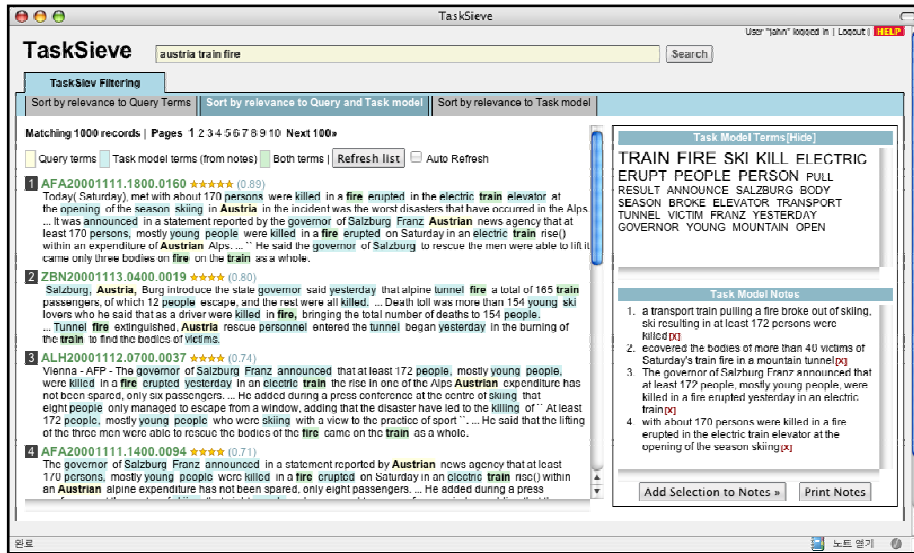


Figure 6.30: TaskSieve system

The goal of the third study was to explore the value of the task model in the information foraging stage of an analyst's work. In order to better support the analyst during a specific task, GALE systems should maintain a task model, which represents the task-related information obtained through interacting with the analyst. TaskSieve was developed as a prototype for GALE systems to explore how to design such a task model and how to employ it to improve user performance.

Ten subjects participated in the TaskSieve study in October, 2007. Each subject was randomly assigned to four tasks selected from EDIE tasks; they each performed two tasks on TaskSieve and the other two on a traditional search system. The sequence of tasks was randomized as well.

The results showed that it is important to integrate a task model for GALE searches. The task model enables a foraging system to keep more relevant items at the top of ranked lists, as compared to Google-like traditional search systems. At the same time, analysts can conduct more productive searches: the average precision values of TaskSieve's ranked lists gathered in the first ten minutes - when users were still relatively unfamiliar with the assigned task - were higher than those of the traditional search system's final ten minutes. Thus, TaskSieve users could select more relevant information earlier in their search sessions.

The results of this study were published by Ahn *et al.* (2008).

6.3.1.3.4. Evaluation of Usages of Named Entities

This study was designed to assess the value of named entities (NE) in supporting both the information foraging and sense-making stages of the analyst's work. In GALE, named entities which refer to people, locations, organizations, events etc. can deliver

important information. NameSieve is a GALE prototype system that supports named entity-based information exploration (Figure 6.31).

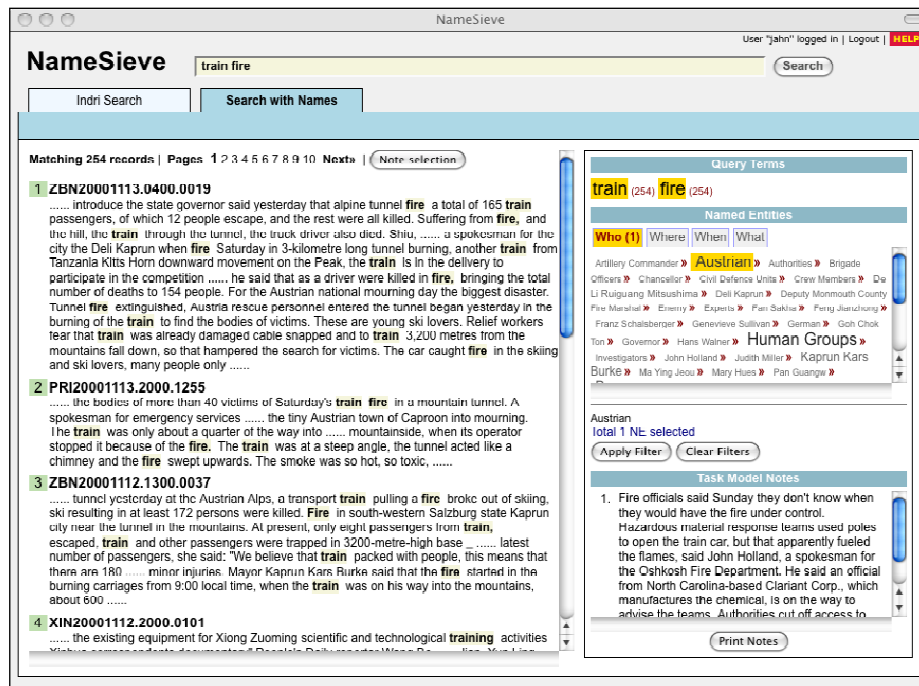


Figure 6.31: NameSieve system

NameSieve provides a simple and intuitive interface to manipulate NEs. Augmenting the conventional ranked list returned by a search engine, NameSieve extracts NEs from the set of retrieved documents and organizes them into tabs corresponding to the “editor’s four-Ws” (Who, What, Where, and When). More frequently occurring names are displayed in larger sizes. Each name can be chosen to serve as a filter via a single mouse click. It allows users to post-filter initial search results and, thus, to focus on information relevant to their tasks.

A named entity tagger developed by IBM was used for the name extraction from the search results. Some heuristics, including Wikipedia article title matching, were used for noise elimination.

Ten subjects participated in the study (from December 2007 to March 2008). The experimental results showed that the NE-based filtering was actively used by the subjects, frequently switching to look for the Who, What, and Where information. The active usage of the NEs led to improvements in the information filtering results. The system performance measured by the precision at ranks five and ten was significantly higher in NameSieve with its NE filters than in the baseline system. Subjects were able to make better annotations for the topic using the NameSieve system. In their feedback, the subjects expressed positive opinions about NameSieve, especially as they accumulated more experience on the search task and had more opportunity to compare the NameSieve with the baseline systems.

6.3.1.4. Discussion and Conclusion

In this contribution, we presented EDIE, an evaluation framework designed for assessing and comparing technologies created for GALE systems. EDIE includes a human-centered, multi-stage, and multi-aspect evaluation methodology; 18 realistic, complex, and dynamic information exploration tasks; and a set of evaluation metrics. A set of EDIE tasks built on the basis of the TDT4 test collection offers passage-level ground truth data annotated on multiple aspects of relevance. All components feature valuable innovations.

Our work brought to light some interesting and important insights related to the evaluation side of the GALE systems. First, the usage of multiple sessions in EDIE was important for capturing the dynamic development of the event and the tasks. According to the results collected from our experiments, this feature indeed showed the complexity of the tasks, the changes in systems performance, and the changes in analysts' behaviors over different sessions.

Second, it was also very helpful to assess separately the information foraging stage and the sense-making stage in GALE searches. This helped to assess how GALE systems are able to support each of these two stages.

Third, our work provided important insights on the evaluation of information exploration systems. Existing information retrieval systems are designed to work well within existing evaluation frameworks - i.e., produce strong topical relevance. However, as our studies hint, topical relevance does not equal better user support. Therefore, this gives us a basis to argue that evaluation involving human users is essential for evaluating GALE systems in order to obtain truly meaningful and useful testing results.

Finally, we observed that the GALE systems' performance may vary a great deal from one task to another. This phenomenon was observed for every type of performance measure explored. Not only do specific performance measures depend upon the specifics of the individual task, but the performance between systems focusing on different information access approaches may fluctuate. This data allows us to argue that instead of perfecting a single specific information access approach such as the Rosetta information filtering, GALE utility systems should use a combination of approaches and engines. Given the large variety of tasks, one engine's advantages can cover the other's weakness in a specific group of tasks.

We hope that this work will contribute to the establishment of evaluation approaches for GALE systems and other similar TBIE systems. We also hope that the EDIE framework will be further utilized to explore various innovations related to user study design and utility evaluation parameters.

6.3.2. Formative Evaluation for Multilingual Multimedia Search and Sense-Making

Authors: Douglas W. Oard, Judith L. Klavans, Dagobert Soergel, Pengyi Zhang, Peter Brusilovsky, Daqing He, Tomasz Loboda, and Leiming Qian

6.3.2.1. Introduction

This section describes the iterative development of Rosetta, a distillation system for multilingual (Arabic, Chinese, English, Spanish) multimedia (television, Web) streaming content. The process involved close collaboration between the IBM T.J. Watson Research Center (IBM), the University of Maryland, the University of Pittsburgh, and Carnegie Mellon University. The fundamental challenge was process-system co-design: as new technical capabilities were introduced, new work processes were sometimes needed to best leverage those capabilities; those new work processes in turn help to identify new technical requirements. The result was a virtuous cycle of innovation.

Rosetta integrates six key technologies to support search and sense-making: Automatic Speech Recognition (ASR), Machine Translation (MT), Information Extraction (IE), Information Retrieval (IR), User Modeling (UM), answer pinpointing for Question Answering (QA), and summary generation from structured knowledge representations. The focus of the work reported in this section was on design innovation and process innovation for integrated architectures (e.g., ASR→MT→IR→IE→QA), in which GALE technologies are used together to accomplish challenging and realistic tasks.

6.3.2.2. The Rosetta System

The Rosetta system consists of three major components: the data collection subsystem, the data processing pipeline, which integrates the ASR, MT and IE components, and the Web application, which integrates IR, UM, QA and summarization components.

Rosetta captures foreign language news broadcasts from Dish Network and performs daily crawl of foreign language Web sites. Rosetta's data processing pipeline is developed on top of IBM's Unstructured Information Management Architecture (UIMA) platform and features a series of data processing components (or "annotators" in UIMA terminology) analyzing input data sequentially. The list of major data annotators in Rosetta includes: ASR, MT, and IE. The Rosetta Web application consists of a browser-based end-user client GUI, and a J2EE back-end server Web application. A browser-based client design was chosen for maximum cross-platform compatibility. Extensive use of Web 2.0 technologies such as AJAX (Asynchronous JavaScript and XML) gives the client a desktop-like feel and enables it to support advanced features not commonly found in conventional Web applications.

Rosetta provides the user with four major modes for information access, each of which can be selected using a tab in the upper left of the screen shots in the following figures. These major modes are organized from left to right in an order that we think of as going from most focused to most general.

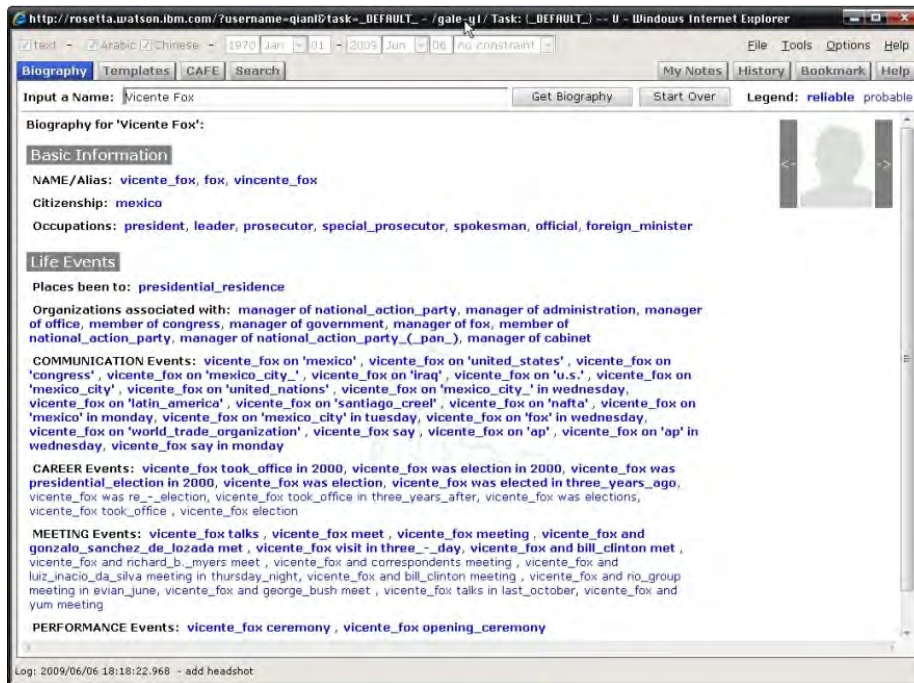


Figure 6.32: The biography mode

The **Biography** and **Templates** modes are driven by the full ASR, MT, IE, QA pipeline. As Figure 6.32 shows, the Biography mode produces an extensive structured display of all information about a person that is known to the system, with drill-down available to a supporting document for each item. In the Template mode, the user can choose from a set of 15 predefined *question templates*, filling in required arguments. The output is a list of snippets that are selected and ordered, based on their likelihood of providing the answer. The **CAFÉ** mode exposes the user to GALE user modeling technology (Yang *et al.* 2007). Rosetta allows each user to create multiple “tasks,” each corresponding to a project that the user is working on over time. The search history, document bookmarks, and notes are all task-specific. When users access documents in Rosetta, their actions are tracked and made visible by the system as “footprints” that can act as reminders. For example, a document that has been visited will be highlighted differently when it shows up again, and users can also select a span of text and add it to the “Notes” tab. Carnegie Mellon University’s Adaptive Filtering Engine (CAFÉ) also uses this evidence to iteratively improve search results. The user first issues a simple query to obtain a list of result snippets from the CAFÉ engine. He can then provide explicit feedback to the CAFÉ engine by rating individual passages as useful (which results in adding it to the notes), irrelevant (which immediately removes it from the view), or redundant (by marking it as “not new”). The CAFÉ engine also accumulates implicit feedback when the user is working in other modes. For example, if a user finds a document using the Search mode and adds some text from this document to his notes, which will be interpreted as “relevant” feedback by CAFÉ.

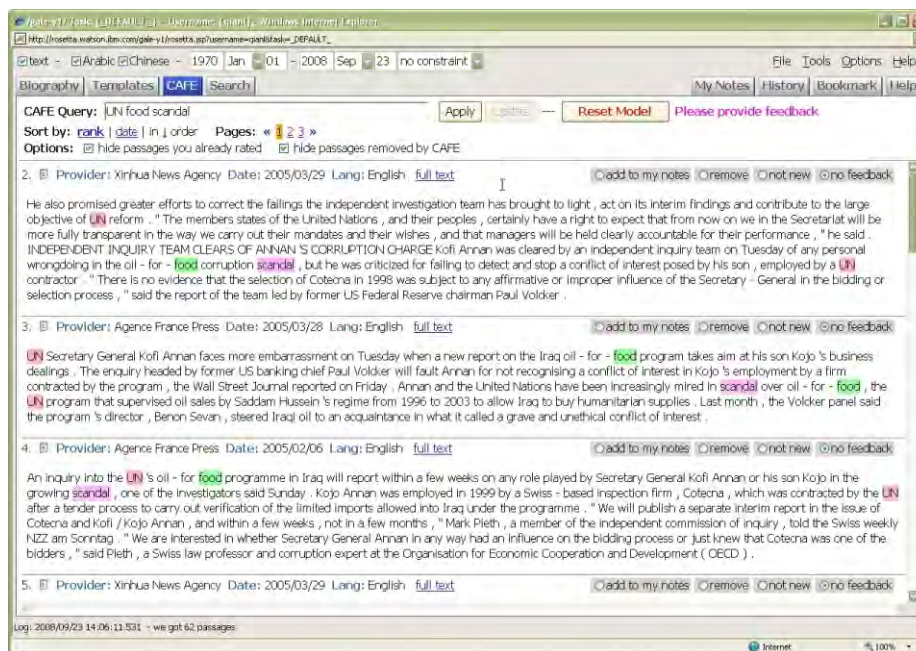


Figure 6.33: The CAFÉ mode

The feedback helps CAFÉ refine the system’s internal model of the user’s task, which can then result in identification of additional useful passages. Figure 6.33 shows the output after just a few feedback iterations. The original query was “UN food scandal;” after adaptive filtering, most of the top passages retrieved are about the food scandal and the role of U.N. secretary Kofi Annan and his son, which in this case are right on target.

The **Search** mode utilizes the GALE ASR, MT, and IR technologies. It allows the user to do conventional keyword search using English queries to find foreign-language multimedia content. Among the four tabs, Search is probably most familiar to most users. For video results, the user can stream the video online with English closed captioning, look at the document in a “Storyboard” type of view, and download the video file together with its captions for local playback. For Web results, the user can look at the non-structured textual content, or submit the cached Web page for on-demand translation.

Rosetta has an elaborate event logging mechanism, able to capture nearly every user action (down to mouse movement and keyboard strokes) and log those events and the associated event context at the server. The events are logged asynchronously in the background so that they do not impact the system’s performance. This functionality makes Rosetta well suited for user studies.

One of the most popular features is the ability to create a Microsoft Word report directly from inside Rosetta. The user can select a span of text or an image and click a button to create a new report in Word, he can further add more material to the report and edit the report freely, and each snippet of information contained in the report also links back to the original document. The user never has to leave Rosetta to compile a report.

6.3.2.3. Formative Evaluation

We conducted 52 formative user study sessions at the University of Maryland and the University of Pittsburgh between June 2006 and May 2007. In this section we describe those studies in four clusters.

Initial Eye-tracking Experiments



Figure 6.34: Heat map showing gaze locations for early version of Templates mode

In an attempt to gain insight into the way users employed the CAFÉ and Templates modes, we used eye tracking to collect user gaze data in several user studies. We employed the Tobii 1750 remote eye-tracking device (<http://www.tobii.com>), which introduced virtually no intrusion in the subject's working environment. The exploratory insights we gained resonated with the more complete and convincing body of evidence described in the following subsections. For example, Figure 6.34 depicts a fixation-count “heat map” for an early version of Templates mode, illustrating a case in which the user may have paid little attention to the control widgets near the top of the screen (in the dark grey “cold” region) because they were placed too far from the focal region (around the Issue Query button in the lower left, where the circular red “hot” region is).

Formative Evaluation for Template QA

In order to evaluate the utility of the Templates mode we ran a controlled experiment that compared it with a Search mode based on the Indri retrieval engine. The study was run at the University of Pittsburgh and involved ten graduate students. As surrogates for intelligence analysts, we recruited study participants who were pursuing advanced degrees in Information Sciences, were native English speakers, and had either 1) taken a masters-level course in information retrieval, or 2) had an extensive background in journalism. Each participant was asked to perform two 40-minute search tasks with Rosetta in a counterbalanced design, one task using only the Search mode and one task

using only the Templates mode. Prior to starting the tasks, each subject received 15 minutes of training on the Templates mode. All subjects were familiar with generic search systems that are similar to the Search mode. Each task required the participant to interact with a large collection of documents to find answers to several questions that had been provided to them in written form. The outcome of their work was a collection of text passages that they had marked as relevant. An analyst could later use these passages to compile a report, but that was not done as a part of this study.

We found that the rate of accumulation of putatively relevant text passages for subjects using Templates was significantly higher than for subjects using Search. That supports the view that it was easier to find relevant pieces of information when using Templates. Further, we found that subjects issued significantly more queries when using Search. As a result, the inter-query time was significantly longer for the Templates mode, which provides evidence that average results of Template search brought more interesting information for users to analyze. Query log analysis indicated that this difference reached a statistically reliable level after 20 minutes. Therefore, for similar studies, we recommend not employing short sessions. We hired three annotators to judge the topical relevance of each text passage collected by our participants. Our comparison of the two modes with respect to the relevance of the results also favored the Templates mode.

Our early version of the Templates mode offered a choice between 16 question types. We found that four question types accounted for 77% of all questions (and that six question types accounted for 89% of all questions). This is an instance of the well-known Pareto (80/20) principle. Of course, the specific questions involved surely depended on the specific tasks that we assigned. Nevertheless, this serves to remind us that identifying the most heavily used functions can help to focus optimization effort.

Two additional formative evaluation studies that were focused on CAFÉ are described in section 6.3.1.3.

Scenario-based Integrated Evaluation

Well designed studies with static collections are particularly useful for comparing the utility of alternative components, but for understanding how complete systems will actually be used we wanted to get as close to the real setting as possible. In our case, this called for additional user studies using live content. We therefore chose a study design in which tasks were designed in near real-time and executed simultaneously by all participants (i.e., without counterbalancing). Our principal goal was to explore process-system co-design: the coupled iteration between new system capabilities and new ways of using the resulting system.

A cohort of information studies graduate students at the University of Maryland was trained as surrogates for intelligence analysts. An initial training session was supplemented with brief lectures at the start of each session on aspects of an analyst's tasks and methods that were relevant to that day's scenario. A second cohort of graduate students performed observational data collection and analysis. Observation notes were automatically integrated with system logs in real time; complementary data were collected from post-session interviews, and both structured questionnaires and brief free-style reaction papers were obtained from participants. Sessions were generally scheduled about two weeks apart, thus allowing some time to implement system improvements.

Summary reports from each session, a requirements tracking database, and weekly teleconferences facilitated information flow between the development and evaluation teams. These studies provided important guidance for user interface refinement, which has been previously reported (Zhang *et al.* 2007). Here, we focus on what was learned about how users actually employ integrated GALE technologies.

Every session that we ran focused principally on Arabic and/or Chinese content (although during training we did make some use of English content). In some cases, the translations were based on ASR transcripts; in other cases the translations were of Web content. We can, therefore, think holistically of our system as a device for assessing the utility of ASR, MT, or ASR-MT cascades for certain tasks. Consistent with previously reported results, we found that cascading present ASR and MT systems often proved to be adequate for tasks that involved identifying topics, sometimes proved to be adequate for tasks that involved detection of specific factual content, and rarely proved to be adequate for tasks that required detection of nuance. We also found that our participants had more difficulty making use of present Chinese MT systems than present Arabic MT systems, which corresponds well to the reported differences in Translation Error Rate (TER) results in machine translation evaluations.

Remarkably often, our participants proved to be adept at using context and (when present) multiple media (e.g., photos or videos) to infer the correct meaning of misrecognized and/or mistranslated terms. For example, several spellings of the same name could be recognized if they appeared next to pictures of the same person. Moreover, our participants learned that using systematically misrecognized and/or mistranslated terms as query terms could sometimes improve results in the Search mode. For example, in a task scenario about Iran's nuclear program, a participant recognized the mistranslated expression “nuclear file” as actually meaning “nuclear program” and used the misrecognized term with good results in subsequent queries. Similarly, a participant recognized “berating women” as the intended meaning of what had been misrecognized as “brainstorming women.”

Our participants were repeatedly observed to spontaneously post-edit incorrect translation results, either using the Notes facility or the Microsoft Word editor in Rosetta. Moreover, when asked to suggest system enhancements, they repeatedly requested the ability to provide feedback to the system about mistranslations. Of course, user study participants may be more focused on the system design, so we do not know whether this preference would also be present in operational users of such systems. Even if the translation system were not adapted, some benefit to future searchers might also accrue from simply memorizing the edits to individual documents. Rosetta now includes this capability.

Earlier user studies had shown that fatigue could become a problem in complex tasks when translations were not easily readable. For this reason, Rosetta includes several features that are designed to help focus the user on specific information (e.g., the snippets in the CAFÉ and Templates mode, and highlighting query terms in Search mode result summaries). These generally proved to be useful, and we believe that investigating additional techniques for helping to guide the user's focus would be useful.

One unexpected and potentially important result was that we repeatedly observed participants engaged in a behavior akin to what journalists might think of as “fact

checking from multiple sources.” In this case, they were not seeking to verify the correctness of the report, but rather the correctness of the translation. Our participants quickly learned that the same pre-translation input typically resulted in the same post-translation output, so they typically dismissed exactly identical output as unhelpful duplication. When they saw the same fact paraphrased in a different way, however, they tended to develop greater confidence in both translations. We are now exploring how we might use this type of cross-source evidence as a basis for unsupervised estimation of translation quality.

We generally left our participants free to use whichever system modes they found most useful, although in some cases we did ask them to use specific modes in order to gain insight into specific usability issues. The Search mode was initially most familiar to our participants, and not surprisingly they initially chose to use it most often. This persisted over time, and an interesting trend emerged: the further to the right the tab was (i.e., the more general the tool), the more often it was used. Such an outcome would not have been predicted from what we had seen earlier in more controlled settings, where Templates and CAFÉ had both shown substantial advantages. Interview and self-report data suggest that the cause was not a reluctance to use new tools; indeed, our participants were all volunteers, and hence could reasonably be described as early adopters of new technologies. Rather, the cause of this effect seems to be twofold. First, more specialized tools are naturally useful in fewer situations. For example, we had a template asking about the relationship between two people (which might be an event), but no template asking about the relationship between two events (which might be a person). Second, our least sophisticated tool (Search) had to expose the most context to the user (for the simple reason that the user had to do more of the job). As our tools became progressively more sophisticated, we adjusted the focus vs. context tradeoff more in favor of focus (although with the full context still available on demand). This became most extreme in the Biography mode, where the initial display included a large number of very short results. With the context less easily accessible, our participants seemed to have more trouble assessing the quality of the system’s results.

Because information seeking is often an iterative process, this difficulty with assessing results may have had the additional effect of making exploratory use of new capabilities somewhat more challenging as well. Indeed, we saw some evidence of that from the way in which participants used CAFÉ. Positive feedback was often provided, and our participants often seemed to be able to develop some degree of confidence from seeing the results of positive feedback. Negative feedback was used far less often, and our users expressed serious reservations about using it in interviews and self-report data. Their reasoning was generally that when they used positive feedback they could see what they were then getting as future results, but when they used negative feedback they had no way of seeing what they then did not get as future results. In this case, the context that they would need extends beyond individual documents – they would also need some way of skimming documents that would have been displayed to them had the negative feedback not been provided.

6.3.2.4. Conclusion and Future Work

Looking back over our full set of studies, we can draw several broad conclusions. More obviously, and most importantly, there are things that we could learn through user studies that we could not have seen as easily (if at all) in “batch-mode” experiments. Just as we use TER as a predictor of Human-assessed Translation Error Rate (HTER) during development, we should be using HTER as a predictor of actual utility during development, but then we should check that prediction from time to time using actual user studies. User studies are more expensive than HTER (which is in turn far more expensive than automatic TER scoring), so we need not do user studies every year. But unless we do user studies periodically, it would be hard to know for sure whether we are headed in the right direction.

Another conclusion that we can draw is that user studies are not all created equal. Highly structured user studies are useful early in a development process when the questions focus on capability, but studies situated in settings that are as representative as possible of the envisioned application are also needed at some point if we are to iterate between system design and discovery of the most effective way to use the resulting systems. Indeed, a sequence of increasingly realistic studies may be needed if we are to optimally balance cost and insight. For example, when actual intelligence analysts later used our systems, we learned that some of the behaviors that we had observed with students (e.g., seeking background information) were less common with actual analysis (who of course started with more background).

Ultimately, the most important benefit of our formative evaluation process may have been the bridges that we built between research communities. Just as GALE has brought speech and translation researchers together to study how best to translate spoken content, we have brought component developers together with system developers, and system developers together with process developers (i.e., real users). After all, the ultimate cascade does not end with transcription, translation, retrieval, extraction, or summarization; it ends with use.

6.3.3. Utility Assessment for the GALE Program

Authors: Connie Fournelle, Jon Holbrook Dan Hunter, and William Salter

6.3.3.1. Introduction

The systems developed under GALE aim to improve timeliness and quality of information available to users via a comprehensive system that supports transcription, translation, and distillation of multiple languages in text and audio formats. The purpose of the Utility Evaluation is to qualitatively and quantitatively assess the performance and utility of the GALE systems in providing these capabilities to the user as he or she interacts with the system. We describe the task definition and environment for the Utility Evaluation, the scenarios used for testing, and the requests for information (RFI's) issued to support them. We further describe how we captured user feedback through written and group feedback sessions, and user performance based on evaluation of final products generated by analysts using GALE systems.

The findings of the user-centric Utility Evaluation complement the findings of the technology-centric Evaluation, which measures the amount of on-target information a GALE system is able to produce in response to a specific, formatted query, and compares GALE system performance to humans completing the same task without GALE tools. The utility of an automated system can be evaluated in terms of the performance of a human user when paired with that system to perform realistic tasks, which may in turn be influenced by a number of factors, including the ease with which that system can be used, as well as, users' desire and willingness to use the system.

6.3.3.2. Utility Evaluation Goals and Methodology

Evaluation Goals

In the context of this evaluation, the term “user” refers to both DoD intelligence analysts and operations personnel. System users were DIA Lead Analysts employed by BAE-Information Technology (BAE -IT) – operational analysts with at least five years of experience with independent project-level responsibility under general supervision – who used the GALE systems during the testing sessions of the evaluation, as well as, the baseline system. The Utility Evaluation was not intended to establish a performance baseline of these analysts, but rather to provide users with exposure to GALE technology, and create a venue for them to give early feedback on the effectiveness of the technology; this feedback can consequently have a direct impact on the research and development activity as it is underway, rather than waiting for the operational needs and research direction to potentially diverge as the program progresses.

Utility Methodology

There were two types of roles for participants in the evaluation: system users and judges. System users were DIA Lead Analysts (mean age = 33 years, SD = 10.0) employed by BAE Systems– operational analysts with an average of 8.7 years of experience (SD = 6.8) performing intelligence-related work – who interacted with the GALE systems during the evaluation. Sixteen system users participated in the Utility Evaluation each day. Thirteen of the system users had previous military experience. All system users were fluent speakers of English; none had any familiarity with spoken or written Arabic or Chinese. The first day of the evaluation familiarized the system users with the goals of the GALE program and the systems under evaluation; on days two-through-five, users employed the GALE systems and a baseline system to generate reports. The reports were evaluated by judges drawn from two different pools of people: BAE Systems principal analysts – analysts with at least eight years of technical leadership including assisting in decision-making in major program operations and supervisory responsibilities – and senior Operations Specialists – operations officers with recent experience in Iraq/Afghanistan. Five judges (mean age = 46 years, SD = 12) participated in the Utility Evaluation (three Principal Analysts and two Operations Specialists).

During the Utility Evaluation, system users were provided with scenarios that described the context, scope, and requirements for the tasks they were to perform using the provided systems. All scenarios were generated in conjunction with or under the guidance of intelligence analysts.

All-source intelligence analysts typically operate within specific domains of expertise (e.g., geo-graphic region, politics, economics, etc.). Because the intelligence analysts serving as system users in the Utility Evaluation were presented with scenarios potentially outside their domains of expertise, each scenario included a section that outlined information related to the scenario topic that the analysts might be assumed to already know if operating in their real working environment.

Each scenario included one or more Requests for Information (RFIs) that define the task(s) that the analysts were asked to perform and the product(s) they were asked to generate. These products, based on those intelligence analysts produce in their everyday jobs, involved writing short (½- to 2- page) prose briefings (e.g., “executive summaries”).

All scenarios were limited to the 1.5-hour testing sessions. That is, a single scenario – with multiple RFIs – was to be completed during each testing session.

Each system user interacted with one system per day over four days. In addition to the three GALE systems under evaluation, users employed a baseline system developed using commercial-off-the-shelf (COTS) technologies, including the desktop search tool Copernic. Users’ work stations included Microsoft Word for the generation of work products. Each of the work stations was connected via a remote connection to offsite software and databases. Connection to other remote sites (e.g., Google, Wikipedia, etc.) was not permitted at any point during the testing sessions. The TDT5 and GALE Y1 corpora (LDC2004E41 and LDC2006E21, respectively), available from the Linguistic Data Consortium, were used in the testing sessions of the Utility Evaluation.

6.3.3.3. Evaluation Measures

Measuring the utility of any system requires identifying ways in which that system aids the user performing her real-world job. Because human behavior in real-world settings is complex, we evaluated a range of quantitative and qualitative measures of performance. The performance measures can be categorized into three broad classes:

- *System-based measures*: these include total time spent in product generation, proportion of time spent in System versus Text Editor (e.g., MS Word), frequency of switches between System and Text Editor, and changes in the ratio of time spent in System versus Text Editor over the time course of product generation.
- *Self-report measures*: All system users participated in debriefing sessions each day of the evaluation that included structured interviews and questionnaires concerning their experience with the system they used that day.
- *Observer-based measures*: Human factors specialists observed users’ interactions with the systems and identified issues with users’ interactions. Judges (described above in Section 2.2) rated each work product on Likert-scale measures of whether the work product was supported by evidence, thorough, organized, relevant, compelling, and actionable. Judges were not provided with information about which users or which GALE systems contributed to a particular work product.

6.3.3.4. Utility Evaluation Results

Product Evaluations

Five judges (i.e., three senior analysts and two military operations specialists) rated each work product using a seven-item instrument that measured the degree to which the work products were supported by evidence, thorough, organized, relevant, compelling, and actionable, as well overall quality. These judges were also asked to rank order all of the products for each RFI from best to worst. All ratings/rankings were performed independently of the other judges, and were performed without any knowledge of which analyst generated the work product or which system the analyst used.

A statistical analysis of the ratings revealed no statistically significant difference (at the level $p < 0.05$) among the systems used for either the simple mean or weighted mean of the ratings of the work products. (The weighted mean combined ratings on highly correlated variables.)

System users were divided into two groups based on their years spent doing intelligence-related work. A median split along this dimension yielded a low experience group ($m = 3.9$ years of intel experience, $SD = 1.5$) and a high experience group ($m = 12.9$ years of intel experience, $SD = 6.9$). Comparisons of overall product ratings for products generated using the Baseline system revealed a marginal effect of system user experience, $F(1,28) = 3.9$, $p = .058$, with products from high experience users (4.23) rated higher than products from low experience users (3.48). However, among GALE systems, no differences were found between ratings of products from high and low experience users, $F < 1$. Given that the Baseline system is similar to the search tools that analysts regularly use, it is not surprising that high experience users were better able to utilize the Baseline system than low experience users. Among the GALE systems, the finding of no effect of user experience on product ratings may indicate a) that the GALE systems function in a way that is sufficiently different from the search tools that analysts regularly use that user experience had little impact of performance in the evaluation, and/or b) the GALE systems provided sufficient support to the less experienced user to boost performance closer to that of more experienced users. This issue is one that may warrant further study in future evaluations.

Usability Evaluations

After completion of each scenario, system users completed a 14-item questionnaire designed to address various aspects of system usability, or the ease with which users felt they were able to interact with the systems. For each item on the questionnaire, system users were asked to rate on a 7-point scale the system they had just used to complete that scenario.

Usefulness Evaluations

After each testing session, participants completed a 12-item questionnaire designed to assess the usefulness of the systems, that is, the degree to which system features would be valuable to the users' job performance. Users were asked to rate the usefulness on 7-point scales of various aspects of the system they had just used. Analysis of overall usefulness ratings revealed a significant effect of System, $F(3,56) = 7.2$, $p < .001$. Two of the GALE

systems were rated higher than Baseline (3.26) with scores of (5.39) and (4.64), $p = .001$ and $p = .02$, respectively.

Overall Results

Question	Low Anchor (I)	Mean	High Anchor (7)
Would be helpful in doing my job	Not helpful	5.77 ^a	Very helpful
Ease of finding info for products	Baseline easier	5.46 ^a	GALE easier
Confidence in finding all available info	Not confident	4.40	Very confident
Ease drill-down/ Follow-up search results	Not easy	5.03 ^a	Very easy
Ease of using results to prepare products	Not easy	5.63 ^a	Very easy
Easier to use to find information	Current system	4.23	GALE
Creates higher workload	Current system	4.07	GALE
Enables finding more relevant info	Baseline	5.54 ^a	GALE
Enables higher quality Product generation	Baseline	5.64 ^a	GALE

Table 6.9: User evaluation of GALE systems (^a means $p < .05$ in comparison with Midpoint of scale).

At the conclusion of testing, system users were asked nine questions designed to capture users' impressions of how the GALE systems might help intelligence analysts perform their jobs, as well as, comparisons of the GALE systems with users' current search tools and with the Baseline system. For the purpose of answering these questions, users were asked to think of the GALE systems in terms of the most effective of the three systems they used during the evaluation.

The results are shown in Table 6.9, below. Responses were on a scale from one-to-seven inclusive, with the low and high anchors indicated for each questions. These results indicated that users believed that GALE technologies would be helpful in doing their jobs, were generally easy to use, and were more effective than the Baseline system. Users rated the GALE systems equivalent to their current systems in terms of ease of use and overall work-load. Users' confidence in the GALE systems to find all available information did not differ from the scale midpoint, which is consistent with findings for this same question from the System Trust instrument, on which the GALE systems were rated as equivalent to or below Baseline. Users' comments about the overall potential of the GALE technologies were generally positive:

“The concept of the systems are really good and should help the open-source community when it's done.”

“All of the systems have features that should be part of the final product.”

A feature of the GALE systems that was frequently mentioned as a valuable feature was the ability to do template directed search in addition to key-word search. However, users also indicated areas for improvement, including expanding the flexibility of the search options, and designing the interface to maximize efficiency of use.

6.3.3.5. Related Work

Gonzalo and Oard (2004) describe a set of experiments conducted by five teams using a common experimental design to test hypotheses about the utility of cross-lingual information retrieval systems. The experiments bear some similarity to the utility evaluation described in this section, but there are important differences.

The experiments are similar to the one described here in that each experiment involved the use of different IR systems by a group of subjects and the answers produced by the subjects were evaluated by an independent group of judges (or “assessors”). Significant differences are that the queries were very specific (e.g., “What year was Thomas Mann awarded the Nobel Prize?”) rather than open-ended as were the queries used in the GALE Utility Evaluation; the systems compared were in general variations on automated IR systems so that a baseline for standard IR retrieval tools such as Google was not established; and finally, the answers produced by the subjects were evaluated on the basis of whether they were correct (valid and supported by documentation), unsupported (valid, but not supported by documentation), or incorrect. The GALE utility evaluation in contrast allowed the judges to additionally express judgments about the degree to which a work product is relevant, well organized, and actionable, and it collected evaluations from the users themselves on their experience with the different systems.

6.3.3.6. Conclusions and Recommendations

We have described the testing framework for the GALE utility evaluation and some of the salient results of that evaluation. In general, the results reflect favorably on the GALE systems, especially in light of users’ initial unfamiliarity with the systems and the limited training time available. We conclude with some recommendations for tool design and future evaluations that are suggested by our analysis of the evaluation results.

In terms of tool design, it would be beneficial to combine key word search capabilities present in current search tools used by analysts with the template-based search capabilities of the GALE systems, since the two kinds of search capabilities turned out to have complementary strengths and weaknesses. Template-based search showed strength when there was a large amount of information relevant to a given query since it allowed for more focused search. When there was little information relevant to a query, however, it proved difficult for users to discover that with template-based search because they could not be sure that all possibilities for filling in template parameters had been exhausted. Having both key word and template-based search capabilities in the same tool would enable analysts to search more efficiently without wasting time on futile searches.

Another design issue that emerged from user feedback was that of tuning the system to adjust the level of irrelevant information returned. Many users expressed distrust in a system if it returned too much irrelevant information. One solution to this problem would be to make the sensitivity of the system an adjustable parameter under the control of the user. Another would be for the system to associate a relevance level with returned information; if the relevance level is strongly correlated with the user's own judgments of relevance, which would increase the user's trust in the system.

Even at the current state of development, analysts rated the GALE systems as useful overall. However, users do not perceive technologies in the same way as technology developers. Specifically, it is not always apparent to system users which aspects of the system are the most difficult to engineer or which aspects of the system are the primary focus of the developer's attention and resources. Rather, users' experience with a system is primarily through its user interface, and fundamental under-the-hood technologies may go completely unnoticed and/or unappreciated by system users. This being the case, there may be a number of low-cost and low-effort features that could be implemented in a GALE tool that, while not the primary focus of the GALE program, can dramatically impact users' satisfaction with and willingness to use the system, which will facilitate transition of the systems into operational settings.

Similarly, when users are evaluating a new technology designed to support or extend their current tasks, to the extent that the tool provides functionality that overlaps with their current tools, users expect the new technology to provide at a minimum all of the functionality of their existing tools in addition to any new features. While this may seem somewhat unreasonable to the developer of the new technology, user satisfaction with any technology designed to replace existing technology requires that the new technology not be viewed as a step backward. Specifically, in the Utility Evaluation users expressed dissatisfaction with the keyword search capabilities of the GALE tools (e.g., no or limited ability to use wildcards). Although keyword search is not the focus of the GALE program, if GALE technologies are intended to replace current keyword search tools, the GALE technologies must be able to perform all of the functions of current keyword search tools. An important distinction to make is that new technologies do not have to instantiate this functionality in the same way as existing technology, but providing that functionality is an important key to successful transition.

Finally, with regard to possible future utility evaluations, user comments on training time and training content were offered frequently and training was mentioned in all four broad categories of the post-scenario surveys (e.g., Workload, Usability, Usefulness, and Trust), as well as, the End-of-Evaluation survey, Daily Debriefs, and Group Debrief. These comments, as well as observations by the evaluation team, indicate clearly that analysts thought they might have been more successful in answering the RFIs had they received more training.

To facilitate training in future utility evaluations, we suggest a more specified approach to the training. Training should include a brief lecture-style introduction, but the majority of the training session should comprise supervised hands-on interaction for the analysts.

References

- Ahn, J., P. Brusilovsky, D. He, J. Grady, and Q. Li. 2008. Personalized Web Exploration with Task Models. *Proceedings of the International World Wide Web Conference*. Beijing, China.
- Allan, J., ed. 2002. *Topic Detection and Tracking: Event-based Information Organization*. Kluwer Academic Publishers.
- Allan, J. 2003. HARD Track Overview in TREC 2003 High Accuracy Retrieval from Documents. *Proceedings of the 12th Text Retrieval Conference*. Gaithersburg, MD.
- Bemish, N. 2008. Broadcast Monitoring Tools Open Doors to Other Cultures, Information. *Communiqué - A Defense Intelligence Agency Publication* 21(2).
- Billa, J., M. Noamany, A. Srivastava, D. Liu, R. Stone, J. Xu, J. Makhoul, and F. Kubala. 2002. Audio Indexing of Arabic Broadcast News. *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing* pp. 15-18. Orlando, FL.
- Chandra, A., D. Kozen, and L. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery* 28(1)pp. 114-133.
- Colbath, S., F. Kubala, D. Liu, and A. Srivastava. 2000. Spoken Documents: Creating Searchable Archives from Continuous Audio. *Proceedings of the 33rd Hawaii International Conference on System Sciences* vol.3. Maui, HI.
- Collins, M. 2002. Discriminative Training Methods For Hidden Markov Models: Theory And Experiments With Perceptron Algorithms. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP) 2002* pp.1-8. Philadelphia, PA.
- Ferrucci, D. and A. Lally. 2004. UIMA: An Architectural Approach to Unstructured Information Processing in the Corporate Research Environment. *Natural Language Engineering* 10(3-4): pp. 327-348.
- Fiscus, J. G. 1997. A Post-Processing System to Yield Reduced Word Error Rates: Recognizer Output Voting Error Reduction (ROVER). *Proceedings of the Institute of Electrical and Electronics Engineers Automatic Speech Recognition and Understanding Workshop* pp. 347-354. Santa Barbara, CA.
- Gilmore, G. J. 2007. Modern Missions Rely on Languages, Cultural Awareness, Official Says. *American Forces Press Service*, April 27.
- Gonzalo, J, and D. W. Oard. 2004. iCLEF 2004 Track Overview: Pilot Experiments in Interactive Cross-Language Question Answering. *Proceedings of the Multilingual Information Access for Text, Speech, and Images, Fifth Workshop of the Cross-Language Evaluation Forum* pp. 310-322. Revised Selected Papers Series, Springer-Verlag, LNCS (3491) pp. 310-322. Bath, UK.

- He, D., and D. Demner-Fushman. 2003. HARD Experiment at Maryland: from Need Negotiation to Automated HARD Process. *Proceedings of the 12th Text Retrieval Conference* pp. 707-714. Gaithersburg, MD.
- He, D., P. Brusilovsky, J. Ahn., J. Grady, R. Farzan, Y. Peng, Y. Yang, and M. Rogati. 2008. An Evaluation of Adaptive Filtering in the Context of Realistic Task-Based Information Exploration. *Information Processing and Management* 44(2): pp. 511-533.
- He, D., P. Brusilovsky, J. Grady, Q. Li, and J-W. Ahn. 2007. How Up-to-date Should It Be? The Value of Instant Profiling and Adaptation in Information Filtering. *Proceedings of the International Conference on Web Intelligence* pp.699-705. Silicon Valley, CA.
- He, D., P. Brusilovsky, J-W. Ahn, J. Grady, R. Farzan, Y. Peng, Y. Yang, and M. Rogati. 2008. An Evaluation of Adaptive Filtering in the Context of Realistic Task-Based Information Exploration. *Information Processing and Management* 44(2): pp. 511-533.
- Huang, J., and G. Zweig. 2002. Maximum Entropy Model for Punctuation Annotation from Speech. *Proceedings of the International Conference on Speech and Language Processing(ICSLP)* pp. 917-920.
- Jayaraman, S., and A. Lavie. 2005. Multi-engine Machine Translation Guided by Explicit Word Matching. *Proceedings of the 10th Annual Conference of the European Association for Machine Translation* pp.143–152. Budapest, Hungary.
- Jones, D. and W. Shen. 2006. Two New Experiments for ILR-Based MT Evaluation. *Proceedings of the Association for Machine Translation in the Americas*. Boston, MA.
- Kubala, F., S. Colbath, D. Liu, A. Srivastava, and J. Makhoul. 2000. Integrated Technologies for Indexing Spoken Language. *Communications of the Association for Computing Machinery* 43: pp. 48–56.
- Kubala, F., S. Colbath, D. Liu, and J. Makhoul. 1999. Rough'n'Ready: a Meeting Recorder and Browser. *Association for Computing Machinery Computing Surveys* 31(2es).
- Liu, D., and F. Kubala. 1999. Fast Speaker Change Detection for Broadcast News Transcription and Indexing. *Proceedings of the 6th European Conference on Speech Communication and Technology (EuroSpeech)* 3: pp. 1031-1034.
- Liu, D., D. Kiecza, A. Srivastava, and F. Kubala. 2005. Online Speaker Adaptation and Tracking for Real-Time Speech Recognition. *Proceedings of the International Speech Communication Association Conference (Interspeech)* pp. 281-284.
- Liu, D., J. Ma, D. Xu, A.Srivastava, and F. Kubala. 2002. Real-Time Rich-Content Transcription Of Chinese Broadcast News. *Proceedings of the International Conference on Speech and Language Processing (ICSLP)* pp. 1981-1984.
- Liu, Y., E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, B. Peskin, and M. Harper. 2004. The ICSI-SRIUW Metadata Extraction System. *Proceedings of the International Conference on Spoken Language Processing* pp. 577-580. Jeju Island, Korea.

- Makhoul, J., F. Kubala, F., T. Leek, D. Liu, L. Nguyen, R. Schwartz, and A. Srivastava. 2000. Speech and Language Technologies For Audio Indexing. *Proceedings of the IEEE* 88: pp. 1338–1353.
- Makhoul, J., F. Kubala, R. Schwartz, and R. Weischedel. 1999. Performance Measures for Information Extraction. *Proceedings of the DARPA Broadcast News Workshop* pp. 249-252. Herndon, VA.
- Marchionini, G., and B. Shneiderman. 1988. Finding Facts Vs. Browsing Knowledge in Hypertext Systems. *IEEE Computer* 21(1): pp. 70-79.
- Matusov, E., A. Mauser, and H. Ney. 2006. Automatic Sentence Segmentation and Punctuation Prediction for Spoken Language Translation. *Proceedings of the International Workshop on Spoken Language Translation (IWSLT)* pp. 158-165. Kyoto, Japan.
- Murthy, U., J. Pitrelli, G. Ramaswamy, M. Franz, and B. Lewis. 2008. A Methodology and Tool Suite for Evaluation of Accuracy of Interoperating Statistical Natural Language Processing Engines. *Proceedings of The International Speech Communication Association Conference (Interspeech)* pp. 2066–2069. Brisbane, Australia.
- Nyberg, E., E. Riebling, R. C. Wang, and R. Frederking. 2008. Integrating a Natural Language Message Pre-processor with UIMA. *Proceedings of LREC Workshop on Enhanced Operability for Large HLT Systems*. Marrakech, Morocco.
- Pirolli, P., and S. Card. 2005. The Sensemaking Process and Leverage Points for Analyst Technology as Identified through Cognitive Task Analysis. *Proceedings of the International Conference on Intelligence Analysis*. McLean, VA.
- Pitrelli, J., B. Lewis, E. Epstein, D. Kiecza, M. Franz, J. Quinn, G. Ramaswamy, A. Srivastava, and P. Virga. 2008a. Aggregating Distributed STT, MT, and Information Extraction Engines: The GALE Interoperability Demo System. *Proceedings of The International Speech Communication Association Conference (INTERSPEECH)* pp. 2743–2746. Brisbane, Australia.
- Pitrelli, J., B. Lewis, E. Epstein, J. Quinn, and G. Ramaswamy. 2008b. A Data Format Enabling Interoperation of Speech Recognition, Translation, and Information Extraction Engines: The GALE Type System. *Proceedings of The International Speech Communication Association Conference (Interspeech)* pp. 1654–1657. Brisbane, Australia.
- Rosti, A., N. Ayan, B. Xiang, S. Matsoukas, R. Schwartz, and B. Dorr. 2007. Combining Outputs from Multiple Machine Translation Systems. *Proceedings of The Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)* pp. 228–235. Rochester, NY.

- Shepard, S., S. Colbath, K. Egan, and F. Kubala. 2002. Oasis Translator's Aide. *Proceedings of the Second international Conference on Human Language Technology Research* pp. 389-390. San Diego, CA.
- Srivastava, A., and F. Kubala. 2003. Sentence Boundary Detection in Arabic Speech. *Proceedings of the European Conference on Speech Communication and Technology (EuroSpeech)* pp. 949-952. Geneva, Switzerland.
- Stolcke, A. and E. Shriberg. 1996. Automatic Linguistic Segmentation of Conversational Speech. *Proceedings of the International Conference on Spoken Language Processing* pp. 1005–1008. Philadelphia, PA.
- Stolcke, A., E. Shriberg, R. Bates, M. Ostendorf, D. Hakkani, M. Plache, M., G. Tur, and Y. Lu. 1998. Automatic Detection of Sentence Boundaries And Disfluencies Based On Recognized Words. *Proceedings of International Conference on Spoken Language Processing (ICLP)* pp. 2247-2250. Sydney, Australia.
- White, R. W., G. Muresan, and G. Marchionini. 2006. Evaluating Exploratory Search Systems. *Proceedings of the Association for Computing Machinery's Special Interest Group on Information Retrieval Conference*. Seattle, WA.
- Yang, Y., A. Lad., N. Lao, A. Harpale, B. Kisiel, M. Rogati. 2007. Utility-based Information Distillation Over Temporally Sequenced Documents. *Proceedings of the International Association for Computing Machinery Special Interest Group on Information Retrieval Conference on Research and Development in Information Retrieval*. pp. 31-38. Amsterdam, Netherlands.
- Zhang, P., E. Plettenberg, J. Klavans, D. Oard, and D. Soergel. 2007. Task-based Interaction with an Integrated Multilingual, Multimedia Information System: A Formative Evaluation. *Proceedings of the Joint Conference on Digital Libraries*. Vancouver, Canada.

Concluding Remarks

Authors: Joseph Olive and Caitlin Christianson

The achievements made in the first three years of the GALE Program have been astonishing. Arabic newswire translation accuracy has increased from an approximate average of 55 percent before GALE to an average of 92 percent at the end of the third year of GALE, and Arabic broadcast news translation accuracy increased from an approximate average of 35 percent to an average of 89 percent during that same period.¹ Chinese translation, a much more difficult task than Arabic translation, was initially far less accurate but has shown a similar trend of improvement. DARPA has also compared the quality of GALE machine translation of text from both structured and unstructured genres to two versions of human translation: a translation created by a single person without subsequent editing, and translation that has been edited by a second translator. The results showed that for Arabic, the translations produced by GALE systems were comparable to the human-generated translations that had not undergone any editing – indicating that GALE Arabic translation was approaching edit-worthy quality. The results for Chinese translation showed that a Chinese-English translation created by a single human generally required more editing than did an Arabic-English translation, which confirmed the difficulty of Chinese-English translation, even for humans. This difficulty is reflected in the fact that while translation of Chinese has improved greatly during the first three years of the GALE Program, at the end of the third year, it is still not at the level of accuracy of an un-edited human translation.

The enormous improvements in machine translation accuracy achieved during the first three years of GALE were enabled by a variety of factors. First of all, GALE was adequately funded, so it was possible to assemble a large, creative, and devoted community. GALE researchers were able to expand on previous research, which consisted mostly of phrase-based translation, with the introduction of many new translation methods, which are described in this book. Some of the most prominent among the new machine translation developments achieved under GALE were use of morphology to reduce the out-of-vocabulary (OOV) rate and help with syntactic analysis, creation of novel algorithms for acoustic modeling, implementation of topic-dependent language models, adoption of long-distance language modes, and integration of hierarchical translation, syntax, and rule induction.

Great achievements have also been made in the GALE Program in the area of distillation. GALE distillation systems have shown a capability to respond to queries producing only relevant information and collapsing all redundancies into integrated responses with citations to all relevant documents. GALE distillation engines are responsive to queries. An engine does not merely produce a response that matches all or some of the words in a query, but is actually aware of word functions (i.e. who did what to whom). Since distillation involves multilingual search, systems are capable of

¹ Pre-GALE baseline should be considering approximate due to the limited amount of data on which the calculation was based.

matching similar documents, even in instances in which a translation is not perfect, working both in the source language and in the target language (English). In addition, distillation queries also assist translation, especially in the case of OOV words, which are often names. The queries actually enable the translation engines to disambiguate words and assign meanings to unfamiliar words.

With distillation quality measured as a percentage relative to human performance on a query basis, our latest results show that for 55 percent of the queries in all languages and genres, GALE distillation systems are able to produce more information than humans and that for more than 70 percent of the queries in English, GALE distillation systems are able to produce more information than humans. Since testing includes reduction of scores for both irrelevant and redundant responses, it is clear that the distillation process could result in a substantial increase in efficiency.

Finally, there is the important question of what will come after GALE. At the GALE kick-off meeting, someone asked what would be left if all GALE targets were to be met. If one observes what has been accomplished under GALE, the answer becomes quite simple. All that is necessary to envision the near future of human language processing at DARPA is to analyze the goals of the GALE Program.

GALE was designed to achieve success in a narrow but extremely important area of human language processing. GALE research was intended to address transcription, translation, and distillation of news material, and has been expanded to address also talk shows, weblogs, and newsgroups. These three additional genres do involve language that is not scripted or edited, but they are far from being truly conversational and informal. Also, GALE was designed to involve analysis of two languages, Arabic and Chinese – both interesting, both extremely different from English and each other, and both important to the U.S. government. There are, however, more than 6,000 languages in the world, at least 200 of which are of interest to the U.S. government. It is not feasible to spend the amount of time and financial resources allocated to the GALE Program to address each of these. Although much of the technology developed through GALE involves language-independent learning from parallel corpora, such data is not available for all language pairs. In addition, in order to achieve the quality of GALE translation, extensive additional data annotation has been necessary. Similarly, in order to improve distillation performance and increase the number of languages handled by distillation, it will be necessary to increase capabilities in syntactic, pragmatic, and semantic analysis.

There is also the issue that while Chinese translation quality has increased under GALE, and it will likely continue to be improved in the final two years of the program, processing of Chinese will likely still require a great deal of improvement. Achieving extremely high quality Chinese-English translation will require additional research and innovation to deal with word segmentation, word order, inherent ambiguities, and other challenges.

After the conclusion of the GALE program, it will next be necessary to tackle the task of addressing informal genre materials, such as e-mail, messaging, and natural conversation for Arabic and Chinese, as well as English. Success at this task will require better handling of dialects, regional accents in speech, incomplete or grammatically-inaccurate sentences, misspelled words, and ambiguous references, among other issues. The tireless efforts of the researchers involved in GALE and previous research programs

have established a solid foundation for dealing with these and other newly-emerging human language processing problems. Additional research will still be necessary to develop means to cope with the difficulties involved in dialect identification, aspects of language modeling, parsing of the types of sentences mentioned above, and more.

Another area for research is development of processing and translation of new languages, especially those for which few resources exist. Even when it is possible to obtain or create large parallel corpora, use of these data tends to result in translation that does not meet the quality standard for usability for many purposes and does not produce output that is optimally useful for search, extraction, information retrieval, and information analysis. To reach the levels that have been attained for Arabic-English translation in the GALE Program, researchers carried out and employed annotation like treebanking, careful word alignment, and predicate argument annotation. Because these types of linguistic data enrichment are extremely expensive and time consuming, additional research is needed to develop means to derive such annotations automatically from accurate and comprehensive bilingual dictionaries, small parallel corpora, and large monolingual corpora in the desired languages. When dealing with speech input, there is also a need to discover ways to adapt the acoustic models from previously-researched languages to new languages. This adaptation is necessary because phoneme sets often differ substantially between languages.

Another way in which GALE technology is limited is that it is passive, in the sense that one reads or listens to information translated and processed from a foreign language. In this mode of operation, errors are transmitted but do not accumulate. In cases where translation is intended to be used for interaction, such as with multi-lingual conversation messaging, and chat, errors can cause misunderstandings or even derail the entire interaction. Obviously, the highest grade of translation is necessary for such purposes, but it would also be advantageous for a translation system to be able to assess its confidence in the accuracy of the translations it produces and construct clarifying dialogues to ensure that the translations are correct.

Such a human-machine dialogue would also be useful for applications other than translation of one human language into another, such as serving as an enabler of command and control management, filling knowledge bases, and translating human languages into computer languages. For these applications, multimodal communication, using modes like eye gaze, gesturing, pointing, drawing, and more, would enhance the communication helping with disambiguation and increasing efficiency.

In addition to the limitations of GALE research with regard to language, genre, and interaction mode, there is also the fact that GALE research involves only broadcast-quality speech transmission signals. To deal with other, more challenging modes of transmission, it will be necessary to expand the capabilities developed under GALE to address the challenges posed by different transmission media and environmental noise factors.

Finally, to achieve full language comprehension, it will be necessary to ground language in the real world. One means of doing so would be to use visual and tactile information to teach a machine the meanings of words so that the machine is able to understand the properties and functions of objects, the consequences of actions, such as the state of the world before and after an action, and what happens during an action and

why. Such knowledge, coupled with the ability to reason, would enable machines to learn language and its subtleties.

The ongoing efforts of GALE researchers have resulted in vast contributions to and will continue to advance language processing research, but GALE is only a beginning.