NAACL HLT 2009

Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics

Short Papers

May 31 – June 5, 2009
Boulder, Colorado

Production and Manufacturing by
Omnipress Inc.
2600 Anderson Street
Madison, WI 53707
USA

Sponsors:

- Rosetta Stone
- CNGL
- Microsoft Research
- Google
- AT&T
- Language Weaver
- J.D. Power
- IBM Research
- The Linguistic Data Consortium
- The Human Language Technology Center of Excellence at the Johns Hopkins University
- The Computational Language and Education Research Center at the University of Colorado at Boulder

Order copies of this and other ACL proceedings from:

Association for Computational Linguistics (ACL)
209 N. Eighth Street
Stroudsburg, PA 18360
USA
Tel: +1-570-476-8006
Fax: +1-570-476-0860
acl@aclweb.org

ISBN: 978-1-932432-42-8

Table of Contents

Conference Program Overview

Monday, June 1, 2009

9:00–10:10 Plenary Session – Invited Talk by Antonio Torralba: *Understanding Visual Scenes*

10:40–11:20 Session 1A: Semantics
Session 1B: Multilingual Processing / Morphology and Phonology
Session 1C: Syntax and Parsing
Student Research Workshop Session 1

2:00–3:30 Short Paper Presentations:
Session 2A: Machine Translation
Session 2B: Information Retrieval / Information Extraction / Sentiment
Session 2C: Dialog / Speech / Semantics
Student Research Workshop Session 2

4:00–5:40 Session 3A: Machine Translation
Session 3B: Semantics
Session 3C: Information Retrieval
Student Research Workshop Session 3

6:30–9:30 Poster and Demo Session
Student Research Workshop Poster Session

Tuesday, June 2, 2009

9:00-10:10 Plenary Session: Paper Award Presentations

10:10–11:40 Session 4A: Machine Translation
Session 4B: Sentiment Analysis / Information Extraction
Session 4C: Machine Learning / Morphology and Phonology

2:00–3:30 Short Paper Presentations:
Session 5A: Machine Translation / Generation / Semantics
Session 5B: Machine Learning / Syntax
Session 5C: SPECIAL SESSION – Speech Indexing and Retrieval

4:00–5:15 Session 6A: Syntax and Parsing
Session 6B: Discourse and Summarization
Session 6C: Spoken Language Systems

Wednesday, June 3, 2009

9:00–10:10	Plenary Session – Invited Talk by Dan Jurafsky: *Ketchup, Espresso, and Chocolate Chip Cookies: Travels in the Language of Food*
10:40–12:20	Session 7A: Machine Translation Session 7B: Speech Recognition and Language Modeling Session 7C: Sentiment Analysis
12:40-1:40	Panel Discussion: *Emerging Application Areas in Computational Linguistics*
1:40–2:30	NAACL Business Meeting
2:30–3:45	Session 8A: Large-scale NLP Session 8B: Syntax and Parsing Session 8C: Discourse and Summarization
4:15–5:30	Session 9A: Machine Learning Session 9B: Dialog Systems Session 9C: Syntax and Parsing

Conference Program

Monday, June 1, 2009

Plenary Session

9:00–10:10 Welcome and Invited Talk: *Understanding Visual Scenes*
Antonio Torralba

10:10–10:40 **Break**

Session 1A: Semantics

Note: all full papers are located in the Main volume of the proceedings

10:40–11:05 *Subjectivity Recognition on Word Senses via Semi-supervised Mincuts*
Fangzhong Su and Katja Markert

11:05–11:30 *Integrating Knowledge for Subjectivity Sense Labeling*
Yaw Gyamfi, Janyce Wiebe, Rada Mihalcea and Cem Akkaya

11:30–11:55 *A Study on Similarity and Relatedness Using Distributional and WordNet-based Approaches*
Eneko Agirre, Enrique Alfonseca, Keith Hall, Jana Kravalova, Marius Pasca and Aitor Soroa

11:55–12:20 *A Fully Unsupervised Word Sense Disambiguation Method Using Dependency Knowledge*
Ping Chen, Wei Ding, Chris Bowes and David Brown

Session 1B: Multilingual Processing / Morphology and Phonology

10:40–11:05 *Learning Phoneme Mappings for Transliteration without Parallel Data*
Sujith Ravi and Kevin Knight

11:05–11:30 *A Corpus-Based Approach for the Prediction of Language Impairment in Monolingual English and Spanish-English Bilingual Children*
Keyur Gabani, Melissa Sherman, Thamar Solorio, Yang Liu, Lisa Bedore and Elizabeth Peña

11:30–11:55 *A Discriminative Latent Variable Chinese Segmenter with Hybrid Word/Character Information*
Xu Sun, Yaozhong Zhang, Takuya Matsuzaki, Yoshimasa Tsuruoka and Jun'ichi Tsujii

11:55–12:20 *Improved Reconstruction of Protolanguage Word Forms*
Alexandre Bouchard-Côté, Thomas L. Griffiths and Dan Klein

Monday, June 1, 2009 (continued)

Session 1C: Syntax and Parsing

10:40–11:05 *Shared Logistic Normal Distributions for Soft Parameter Tying in Unsupervised Grammar Induction*
Shay Cohen and Noah A. Smith

11:05–11:30 *Adding More Languages Improves Unsupervised Multilingual Part-of-Speech Tagging: a Bayesian Non-Parametric Approach*
Benjamin Snyder, Tahira Naseem, Jacob Eisenstein and Regina Barzilay

11:30–11:55 *Efficiently Parsable Extensions to Tree-Local Multicomponent TAG*
Rebecca Nesson and Stuart Shieber

11:55–12:20 *Improving Unsupervised Dependency Parsing with Richer Contexts and Smoothing*
William P. Headden III, Mark Johnson and David McClosky

Student Research Workshop Session 1:

Note: all student research workshop papers are located in the Companion volume of the proceedings

10:40–11:10 *Classifier Combination Techniques Applied to Coreference Resolution*
Smita Vemulapalli, Xiaoqiang Luo, John F. Pitrelli and Imed Zitouni

11:15–11:45 *Solving the "Who's Mark Johnson Puzzle": Information Extraction Based Cross Document Coreference*
Jian Huang, Sarah M. Taylor, Jonathan L. Smith, Konstantinos A. Fotiadis and C. Lee Giles

11:50–12:20 *Exploring Topic Continuation Follow-up Questions using Machine Learning*
Manuel Kirschner and Raffaella Bernardi

12:20–2:00 **Lunch Break**

Monday, June 1, 2009 (continued)

Session 2A: Short Paper Presentations: Machine Translation

2:00–2:15 *Cohesive Constraints in A Beam Search Phrase-based Decoder*
Nguyen Bach, Stephan Vogel and Colin Cherry

2:15–2:30 *Revisiting Optimal Decoding for Machine Translation IBM Model 4*
Sebastian Riedel and James Clarke

2:30–2:45 *Efficient Extraction of Oracle-best Translations from Hypergraphs*
Zhifei Li and Sanjeev Khudanpur

2:45–3:00 *Semantic Roles for SMT: A Hybrid Two-Pass Model*
Dekai Wu and Pascale Fung

3:00–3:15 *Comparison of Extended Lexicon Models in Search and Rescoring for SMT*
Saša Hasan and Hermann Ney

3:15–3:30 *A Simplex Armijo Downhill Algorithm for Optimizing Statistical Machine Translation Decoding Parameters*
Bing Zhao and Shengyuan Chen

Session 2B: Short Paper Presentations: Information Retrieval / Information Extraction / Sentiment

2:00–2:15 *Translation Corpus Source and Size in Bilingual Retrieval*
Paul McNamee, James Mayfield and Charles Nicholas

2:15–2:30 *Large-scale Computation of Distributional Similarities for Queries*
Enrique Alfonseca, Keith Hall and Silvana Hartmann

2:30–2:45 *Text Categorization from Category Name via Lexical Reference*
Libby Barak, Ido Dagan and Eyal Shnarch

2:45–3:00 *Identifying Types of Claims in Online Customer Reviews*
Shilpa Arora, Mahesh Joshi and Carolyn P. Rosé

3:00–3:15 *Towards Automatic Image Region Annotation - Image Region Textual Coreference Resolution*
Emilia Apostolova and Dina Demner-Fushman

Monday, June 1, 2009 (continued)

3:15–3:30 *TESLA: A Tool for Annotating Geospatial Language Corpora*
Nate Blaylock, Bradley Swain and James Allen

Session 2C: Short Paper Presentations: Dialog / Speech / Semantics

2:00–2:15 *Modeling Dialogue Structure with Adjacency Pair Analysis and Hidden Markov Models*
Kristy Elizabeth Boyer, Robert Phillips, Eun Young Ha, Michael Wallis, Mladen Vouk and James Lester

2:15–2:30 *Towards Natural Language Understanding of Partial Speech Recognition Results in Dialogue Systems*
Kenji Sagae, Gwen Christian, David DeVault and David Traum

2:30–2:45 *Spherical Discriminant Analysis in Semi-supervised Speaker Clustering*
Hao Tang, Stephen Chu and Thomas Huang

2:45–3:00 *Learning Bayesian Networks for Semantic Frame Composition in a Spoken Dialog System*
Marie-Jean Meurs, Fabrice Lefèvre and Renato De Mori

3:00–3:15 *Evaluation of a System for Noun Concepts Acquisition from Utterances about Images (SINCA) Using Daily Conversation Data*
Yuzu Uchida and Kenji Araki

3:15–3:30 *Web and Corpus Methods for Malay Count Classifier Prediction*
Jeremy Nicholson and Timothy Baldwin

Monday, June 1, 2009 (continued)

Student Research Workshop Session 2

Note: all student research workshop papers are located in the Companion volume of the proceedings

2:00–2:30 *Sentence Realisation from Bag of Words with Dependency Constraints*
Karthik Gali and Sriram Venkatapathy

2:35–3:05 *Using Language Modeling to Select Useful Annotation Data*
Dmitriy Dligach and Martha Palmer

3:30–4:00 **Break**

Session 3A: Machine Translation

4:00–4:25 *Context-Dependent Alignment Models for Statistical Machine Translation*
Jamie Brunning, Adrià de Gispert and William Byrne

4:25–4:50 *Graph-based Learning for Statistical Machine Translation*
Andrei Alexandrescu and Katrin Kirchhoff

4:50–5:15 *Intersecting Multilingual Data for Faster and Better Statistical Translations*
Yu Chen, Martin Kay and Andreas Eisele

5:15–5:40 No Presentation

Session 3B: Semantics

4:00–4:25 *Without a 'doubt'? Unsupervised Discovery of Downward-Entailing Operators*
Cristian Danescu-Niculescu-Mizil, Lillian Lee and Richard Ducott

4:25–4:50 *The Role of Implicit Argumentation in Nominal SRL*
Matthew Gerber, Joyce Chai and Adam Meyers

4:50–5:15 *Jointly Identifying Predicates, Arguments and Senses using Markov Logic*
Ivan Meza-Ruiz and Sebastian Riedel

5:15–5:40 *Structured Generative Models for Unsupervised Named-Entity Clustering*
Micha Elsner, Eugene Charniak and Mark Johnson

Monday, June 1, 2009 (continued)

Session 3C: Information Retrieval

4:00–4:25 *Hierarchical Dirichlet Trees for Information Retrieval*
Gholamreza Haffari and Yee Whye Teh

4:25–4:50 *Phrase-Based Query Degradation Modeling for Vocabulary-Independent Ranked Utterance Retrieval*
J. Scott Olsson and Douglas W. Oard

4:50–5:15 *Japanese Query Alteration Based on Lexical Semantic Similarity*
Masato Hagiwara and Hisami Suzuki

5:15–5:40 *Context-based Message Expansion for Disentanglement of Interleaved Text Conversations*
Lidan Wang and Douglas Oard

Student Research Workshop Session 3

Note: all student research workshop papers are located in the Companion volume of the proceedings

4:00–4:30 *Pronunciation Modeling in Spelling Correction for Writers of English as a Foreign Language*
Adriane Boyd

4:35–5:05 *Building a Semantic Lexicon of English Nouns via Bootstrapping*
Ting Qian, Benjamin Van Durme and Lenhart Schubert

5:10–5:40 *Multiple Word Alignment with Profile Hidden Markov Models*
Aditya Bhargava and Grzegorz Kondrak

6:30–9:30 **Poster and Demo Session**

Note: all demo abstracts are located in the Companion volume of the proceedings

Minimum Bayes Risk Combination of Translation Hypotheses from Alternative Morphological Decompositions
Adrià de Gispert, Sami Virpioja, Mikko Kurimo and William Byrne

Generating Synthetic Children's Acoustic Models from Adult Models
Andreas Hagen, Bryan Pellom and Kadri Hacioglu

Monday, June 1, 2009 (continued)

Detecting Pitch Accents at the Word, Syllable and Vowel Level
Andrew Rosenberg and Julia Hirschberg

Shallow Semantic Parsing for Spoken Language Understanding
Bonaventura Coppola, Alessandro Moschitti and Giuseppe Riccardi

Automatic Agenda Graph Construction from Human-Human Dialogs using Clustering Method
Cheongjae Lee, Sangkeun Jung, Kyungduk Kim and Gary Geunbae Lee

A Simple Sentence-Level Extraction Algorithm for Comparable Data
Christoph Tillmann and Jian-ming Xu

Learning Combination Features with L1 Regularization
Daisuke Okanohara and Jun'ichi Tsujii

Multi-scale Personalization for Voice Search Applications
Daniel Bolaños, Geoffrey Zweig and Patrick Nguyen

The Importance of Sub-Utterance Prosody in Predicting Level of Certainty
Heather Pon-Barry and Stuart Shieber

Using Integer Linear Programming for Detecting Speech Disfluencies
Kallirroi Georgila

Contrastive Summarization: An Experiment with Consumer Reviews
Kevin Lerman and Ryan McDonald

Topic Identification Using Wikipedia Graph Centrality
Kino Coursey and Rada Mihalcea

Extracting Bilingual Dictionary from Comparable Corpora with Dependency Heterogeneity
Kun Yu and Jun'ichi Tsujii

Domain Adaptation with Artificial Data for Semantic Parsing of Speech
Lonneke van der Plas, James Henderson and Paola Merlo

Monday, June 1, 2009 (continued)

Extending Pronunciation Lexicons via Non-phonemic Respellings
Lucian Galescu

A Speech Understanding Framework that Uses Multiple Language Models and Multiple Understanding Models
Masaki Katsumaru, Mikio Nakano, Kazunori Komatani, Kotaro Funakoshi, Tetsuya Ogata and Hiroshi G. Okuno

Taking into Account the Differences between Actively and Passively Acquired Data: The Case of Active Learning with Support Vector Machines for Imbalanced Datasets
Michael Bloodgood and Vijay Shanker

Faster MT Decoding Through Pervasive Laziness
Michael Pust and Kevin Knight

Evaluating the Syntactic Transformations in Gold Standard Corpora for Statistical Sentence Compression
Naman K. Gupta, Sourish Chaudhuri and Carolyn P. Rosé

Incremental Adaptation of Speech-to-Speech Translation
Nguyen Bach, Roger Hsiao, Matthias Eck, Paisarn Charoenpornsawat, Stephan Vogel, Tanja Schultz, Ian Lane, Alex Waibel and Alan Black

Name Perplexity
Octavian Popescu

Answer Credibility: A Language Modeling Approach to Answer Validation
Protima Banerjee and Hyoil Han

Exploiting Named Entity Classes in CCG Surface Realization
Rajakrishnan Rajkumar, Michael White and Dominic Espinosa

Search Engine Adaptation by Feedback Control Adjustment for Time-sensitive Query
Ruiqiang Zhang, Yi Chang, Zhaohui Zheng, Donald Metzler and Jian-yun Nie

A Local Tree Alignment-based Soft Pattern Matching Approach for Information Extraction
Seokhwan Kim, Minwoo Jeong and Gary Geunbae Lee

Classifying Factored Genres with Part-of-Speech Histograms
Sergey Feldman, Marius Marin, Julie Medero and Mari Ostendorf

Towards Effective Sentence Simplification for Automatic Processing of Biomedical Text
Siddhartha Jonnalagadda, Luis Tari, Jörg Hakenberg, Chitta Baral and Graciela Gonzalez

Improving SCL Model for Sentiment-Transfer Learning
Songbo Tan and Xueqi Cheng

MICA: A Probabilistic Dependency Parser Based on Tree Insertion Grammars (Application Note)
Srinivas Bangalore, Pierre Boullier, Alexis Nasr, Owen Rambow and Benoît Sagot

Lexical and Syntactic Adaptation and Their Impact in Deployed Spoken Dialog Systems
Svetlana Stoyanchev and Amanda Stent

Analysing Recognition Errors in Unlimited-Vocabulary Speech Recognition
Teemu Hirsimäki and Mikko Kurimo

The independence of dimensions in multidimensional dialogue act annotation
Volha Petukhova and Harry Bunt

Improving Coreference Resolution by Using Conversational Metadata
Xiaoqiang Luo, Radu Florian and Todd Ward

Using N-gram based Features for Machine Translation System Combination
Yong Zhao and Xiaodong He

Language Specific Issue and Feature Exploration in Chinese Event Extraction
Zheng Chen and Heng Ji

Improving A Simple Bigram HMM Part-of-Speech Tagger by Latent Annotation and Self-Training
Zhongqiang Huang, Vladimir Eidelman and Mary Harper

6:30–9:30 **Student Research Workshop Poster Session**

Note: all student research workshop papers are located in the Companion volume of the proceedings

Also: All papers presented in the morning and afternoon sessions of the student research workshop will also be shown as posters.

Monday, June 1, 2009 (continued)

Using Emotion to Gain Rapport in a Spoken Dialog System
Jaime Acosta

Interactive Annotation Learning with Indirect Feature Voting
Shilpa Arora and Eric Nyberg

Loss-Sensitive Discriminative Training of Machine Transliteration Models
Kedar Bellare, Koby Crammer and Dayne Freitag

Syntactic Tree-based Relation Extraction Using a Generalization of Collins and Duffy Convolution Tree Kernel
Mahdy Khayyamian, Seyed Abolghasem Mirroshandel and Hassan Abolhassani

Towards Building a Competitive Opinion Summarization System: Challenges and Keys
Elena Lloret, Alexandra Balahur, Manuel Palomar and Andres Montoyo

Domain-Independent Shallow Sentence Ordering
Thade Nahnsen

Towards Unsupervised Recognition of Dialogue Acts
Nicole Novielli and Carlo Strapparava

Modeling Letter-to-Phoneme Conversion as a Phrase Based Statistical Machine Translation Problem with Minimum Error Rate Training
Taraka Rama, Anil Kumar Singh and Sudheer Kolachina

Disambiguation of Preposition Sense Using Linguistically Motivated Features
Stephen Tratz and Dirk Hovy

Tuesday, June 2, 2009

Plenary Session

9:00–9:10 Paper Awards

9:10–9:40 *Unsupervised Morphological Segmentation with Log-Linear Models*
Hoifung Poon, Colin Cherry and Kristina Toutanova

9:40–10:10 *11,001 New Features for Statistical Machine Translation*
David Chiang, Kevin Knight and Wei Wang

10:10-10:40 **Break**

Session 4A: Machine Translation

10:10–10:35 *Efficient Parsing for Transducer Grammars*
John DeNero, Mohit Bansal, Adam Pauls and Dan Klein

10:35–10:50 *Preference Grammars: Softening Syntactic Constraints to Improve Statistical Machine Translation*
Ashish Venugopal, Andreas Zollmann, Noah Smith and Stephan Vogel

10:50–11:15 *Using a Dependency Parser to Improve SMT for Subject-Object-Verb Languages*
Peng Xu, Jaeho Kang, Michael Ringgaard and Franz Och

11:15–11:40 *Learning Bilingual Linguistic Reordering Model for Statistical Machine Translation*
Han-Bin Chen, Jian-Cheng Wu and Jason S. Chang

Tuesday, June 2, 2009 (continued)

Session 4B: Sentiment Analysis / Information Extraction

10:10–10:35 *May All Your Wishes Come True: A Study of Wishes and How to Recognize Them*
Andrew Goldberg, Nathanael Fillmore, David Andrzejewski, Zhiting Xu, Bryan Gibson and Xiaojin Zhu

10:35–10:50 *Predicting Risk from Financial Reports with Regression*
Shimon Kogan, Dimitry Levin, Bryan R. Routledge, Jacob S. Sagi and Noah A. Smith

10:50–11:15 *Domain Adaptation with Latent Semantic Association for Named Entity Recognition*
Honglei Guo, Huijia Zhu, Zhili Guo, Xiaoxun Zhang, Xian Wu and Zhong Su

11:15–11:40 *Semi-Automatic Entity Set Refinement*
Vishnu Vyas and Patrick Pantel

Session 4C: Machine Learning / Morphology and Phonology

10:10–10:35 *Unsupervised Constraint Driven Learning For Transliteration Discovery*
Ming-Wei Chang, Dan Goldwasser, Dan Roth and Yuancheng Tu

10:35–10:50 *On the Syllabification of Phonemes*
Susan Bartlett, Grzegorz Kondrak and Colin Cherry

10:50–11:15 *Improving nonparameteric Bayesian inference: experiments on unsupervised word segmentation with adaptor grammars*
Mark Johnson and Sharon Goldwater

11:15–11:40 No Presentation

12:20–2:00 **Lunch Break**

Tuesday, June 2, 2009 (continued)

Session 5A: Short Paper Presentations: Machine Translation / Generation / Semantics

2:00–2:15 *Statistical Post-Editing of a Rule-Based Machine Translation System*
Antonio-L. Lagarda, Vicent Alabau, Francisco Casacuberta, Roberto Silva and Enrique Díaz-de-Liaño

2:15–2:30 *On the Importance of Pivot Language Selection for Statistical Machine Translation*
Michael Paul, Hirofumi Yamamoto, Eiichiro Sumita and Satoshi Nakamura

2:30–2:45 *Tree Linearization in English: Improving Language Model Based Approaches*
Katja Filippova and Michael Strube

2:45–3:00 *Determining the position of adverbial phrases in English*
Huayan Zhong and Amanda Stent

3:00–3:15 *Estimating and Exploiting the Entropy of Sense Distributions*
Peng Jin, Diana McCarthy, Rob Koeling and John Carroll

3:15–3:30 *Semantic Classification with WordNet Kernels*
Diarmuid Ó Séaghdha

Session 5B: Short Paper Presentations: Machine Learning / Syntax

2:00–2:15 *Sentence Boundary Detection and the Problem with the U.S.*
Dan Gillick

2:15–2:30 *Quadratic Features and Deep Architectures for Chunking*
Joseph Turian, James Bergstra and Yoshua Bengio

2:30–2:45 *Active Zipfian Sampling for Statistical Parser Training*
Onur Çobanoğlu

2:45–3:00 *Combining Constituent Parsers*
Victoria Fossum and Kevin Knight

3:00–3:15 *Recognising the Predicate-argument Structure of Tagalog*
Meladel Mistica and Timothy Baldwin

Tuesday, June 2, 2009 (continued)

3:15–3:30 *Reverse Revision and Linear Tree Combination for Dependency Parsing*
Giuseppe Attardi and Felice Dell'Orletta

Session 5C: Short Paper Presentations: SPECIAL SESSION – Speech Indexing and Retrieval

2:00–2:15 *Introduction to the Special Session on Speech Indexing and Retrieval*

2:15–2:30 *Anchored Speech Recognition for Question Answering*
Sibel Yaman, Gokan Tur, Dimitra Vergyri, Dilek Hakkani-Tur, Mary Harper and Wen Wang

2:30–2:45 *Score Distribution Based Term Specific Thresholding for Spoken Term Detection*
Dogan Can and Murat Saraclar

2:45–3:00 *Automatic Chinese Abbreviation Generation Using Conditional Random Field*
Dong Yang, Yi-Cheng Pan and Sadaoki Furui

3:00–3:15 *Fast decoding for open vocabulary spoken term detection*
Bhuvana Ramabhadran, Abhinav Sethy, Jonathan Mamou, Brian Kingsbury and Upendra Chaudhari

3:15–3:30 *Tightly coupling Speech Recognition and Search*
Taniya Mishra and Srinivas Bangalore

3:30–4:00 **Break**

Session 6A: Syntax and Parsing

4:00–4:25 *Joint Parsing and Named Entity Recognition*
Jenny Rose Finkel and Christopher D. Manning

4:25–4:50 *Minimal-length linearizations for mildly context-sensitive dependency trees*
Y. Albert Park and Roger Levy

4:50–5:15 *Positive Results for Parsing with a Bounded Stack using a Model-Based Right-Corner Transform*
William Schuler

Tuesday, June 2, 2009 (continued)

Session 6B: Discourse and Summarization

4:00–4:25 *Hierarchical Text Segmentation from Multi-Scale Lexical Cohesion*
Jacob Eisenstein

4:25–4:50 *Exploring Content Models for Multi-Document Summarization*
Aria Haghighi and Lucy Vanderwende

4:50–5:15 *Global Models of Document Structure using Latent Permutations*
Harr Chen, S.R.K. Branavan, Regina Barzilay and David R. Karger

Session 6C: Spoken Language Systems

4:00–4:25 *Assessing and Improving the Performance of Speech Recognition for Incremental Systems*
Timo Baumann, Michaela Atterer and David Schlangen

4:25–4:50 *Geo-Centric Language Models for Local Business Voice Search*
Amanda Stent, Ilija Zeljkovic, Diamantino Caseiro and Jay Wilpon

4:50–5:15 *Improving the Arabic Pronunciation Dictionary for Phone and Word Recognition with Linguistically-Based Pronunciation Rules*
Fadi Biadsy, Nizar Habash and Julia Hirschberg

Wednesday, June 3, 2009

Plenary Session

9:00–10:10 Invited Talk: *Ketchup, Espresso, and Chocolate Chip Cookies: Travels in the Language of Food*
Dan Jurafsky

10:10–10:40 **Break**

Wednesday, June 3, 2009 (continued)

Session 7A: Machine Translation

10:40–11:05 *Using a maximum entropy model to build segmentation lattices for MT*
Chris Dyer

11:05–11:30 *Active Learning for Statistical Phrase-based Machine Translation*
Gholamreza Haffari, Maxim Roy and Anoop Sarkar

11:30–11:55 *Semi-Supervised Lexicon Mining from Parenthetical Expressions in Monolingual Web Pages*
Xianchao Wu, Naoaki Okazaki and Jun'ichi Tsujii

11:55–12:20 *Hierarchical Phrase-Based Translation with Weighted Finite State Transducers*
Gonzalo Iglesias, Adrià de Gispert, Eduardo R. Banga and William Byrne

Session 7B: Speech Recognition and Language Modeling

10:40–11:05 *Improved pronunciation features for construct-driven assessment of non-native spontaneous speech*
Lei Chen, Klaus Zechner and Xiaoming Xi

11:05–11:30 *Performance Prediction for Exponential Language Models*
Stanley Chen

11:30–11:55 *Tied-Mixture Language Modeling in Continuous Space*
Ruhi Sarikaya, Mohamed Afify and Brian Kingsbury

11:55–12:20 *Shrinking Exponential Language Models*
Stanley Chen

Wednesday, June 3, 2009 (continued)

Session 7C: Sentiment Analysis

10:40–11:05 *Predicting Response to Political Blog Posts with Topic Models*
Tae Yano, William W. Cohen and Noah A. Smith

11:05–11:30 *An Iterative Reinforcement Approach for Fine-Grained Opinion Mining*
Weifu Du and Songbo Tan

11:30–11:55 *For a few dollars less: Identifying review pages sans human labels*
Luciano Barbosa, Ravi Kumar, Bo Pang and Andrew Tomkins

11:55–12:20 *More than Words: Syntactic Packaging and Implicit Sentiment*
Stephan Greene and Philip Resnik

12:20–1:40 **Lunch Break**

12:40-1:40 Panel Discussion: *Emerging Application Areas in Computational Linguistics*
Chaired by Bill Dolan, Microsoft
Panelists: Jill Burstein, Educational Testing Service; Joel Tetreault, Educational Testing Service; Patrick Pantel, Yahoo; Andy Hickl, Language Computer Corporation + Swingly

1:40–2:30 NAACL Business Meeting

Session 8A: Large-scale NLP

2:30–2:55 *Streaming for large scale NLP: Language Modeling*
Amit Goyal, Hal Daume III and Suresh Venkatasubramanian

2:55–3:20 *The Effect of Corpus Size on Case Frame Acquisition for Discourse Analysis*
Ryohei Sasano, Daisuke Kawahara and Sadao Kurohashi

3:20–3:45 *Semantic-based Estimation of Term Informativeness*
Kirill Kireyev

Wednesday, June 3, 2009 (continued)

Session 8B: Syntax and Parsing

2:30–2:55 *Optimal Reduction of Rule Length in Linear Context-Free Rewriting Systems*
Carlos Gómez-Rodríguez, Marco Kuhlmann, Giorgio Satta and David Weir

2:55–3:20 *Inducing Compact but Accurate Tree-Substitution Grammars*
Trevor Cohn, Sharon Goldwater and Phil Blunsom

3:20–3:45 *Hierarchical Search for Parsing*
Adam Pauls and Dan Klein

Session 8C: Discourse and Summarization

2:30–2:55 *An effective Discourse Parser that uses Rich Linguistic Information*
Rajen Subba and Barbara Di Eugenio

2:55–3:20 *Graph-Cut-Based Anaphoricity Determination for Coreference Resolution*
Vincent Ng

3:20–3:45 *Using Citations to Generate surveys of Scientific Paradigms*
Saif Mohammad, Bonnie Dorr, Melissa Egan, Ahmed Hassan, Pradeep Muthukrishan, Vahed Qazvinian, Dragomir Radev and David Zajic

3:45–4:15 **Break**

Session 9A: Machine Learning

4:15–4:40 *Non-Parametric Bayesian Areal Linguistics*
Hal Daume III

4:40–5:05 *Hierarchical Bayesian Domain Adaptation*
Jenny Rose Finkel and Christopher D. Manning

5:05–5:30 *Online EM for Unsupervised Models*
Percy Liang and Dan Klein

Wednesday, June 3, 2009 (continued)

Session 9B: Dialog Systems

4:15–4:40 *Unsupervised Approaches for Automatic Keyword Extraction Using Meeting Transcripts*
Feifan Liu, Deana Pennell, Fei Liu and Yang Liu

4:40–5:05 *A Finite-State Turn-Taking Model for Spoken Dialog Systems*
Antoine Raux and Maxine Eskenazi

5:05–5:30 *Extracting Social Meaning: Identifying Interactional Style in Spoken Conversation*
Dan Jurafsky, Rajesh Ranganath and Dan McFarland

Session 9C: Syntax and Parsing

4:15–4:40 *Linear Complexity Context-Free Parsing Pipelines via Chart Constraints*
Brian Roark and Kristy Hollingshead

4:40–5:05 *Improved Syntactic Models for Parsing Speech with Repairs*
Tim Miller

5:05–5:30 *A model of local coherence effects in human sentence processing as consequences of updates from bottom-up prior to posterior beliefs*
Klinton Bicknell and Roger Levy

Cohesive Constraints in A Beam Search Phrase-based Decoder

Nguyen Bach and Stephan Vogel
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{nbach, stephan.vogel}@cs.cmu.edu

Colin Cherry
Microsoft Research
One Microsoft Way
Redmond, WA, 98052, USA
collinc@microsoft.com

Abstract

Cohesive constraints allow the phrase-based decoder to employ arbitrary, non-syntactic phrases, and encourage it to translate those phrases in an order that respects the source dependency tree structure. We present extensions of the cohesive constraints, such as exhaustive interruption count and rich interruption check. We show that the cohesion-enhanced decoder significantly outperforms the standard phrase-based decoder on English→Spanish. Improvements between 0.5 and 1.2 BLEU point are obtained on English→Iraqi system.

1 Introduction

Phrase-based machine translation is driven by a phrasal translation model, which relates phrases (contiguous segments of words) in the source to phrases in the target. This translation model can be derived from a word-aligned bitext. Translation candidates are scored according to a linear model combining several informative feature functions. Crucially, this model incorporates translation model scores and n-gram language model scores. The component features are weighted to minimize a translation error criterion on a development set (Och, 2003). Decoding the source sentence takes the form of a beam search through the translation space, with intermediate states corresponding to partial translations. The decoding process advances by extending a state with the translation of a source phrase, until each source word has been translated exactly once. Re-ordering occurs when the source phrase to be translated does not immediately follow the previously translated phrase. This is penalized with a discriminatively-trained distortion penalty. In order to calculate the current translation score, each state can be represented by a triple:

- A coverage vector HC indicates which source words have already been translated.
- A span $\bar{f}$ indicates the last source phrase translated to create this state.
- A target word sequence stores context needed by the target language model.

As cohesion concerns only movement in the source, we can completely ignore the language model context, making state effectively an $(\bar{f}, HC)$ tuple.

To enforce cohesion during the state expansion process, cohesive phrasal decoding has been proposed in (Cherry, 2008; Yamamoto et al., 2008). The cohesion-enhanced decoder enforces the following constraint: once the decoder begins translating any part of a source subtree, it must cover all the words under that subtree before it can translate anything outside of it. This notion can be applied to any projective tree structure, but we use dependency trees, which have been shown to demonstrate greater cross-lingual cohesion than other structures (Fox, 2002). We use a tree data structure to store the dependency tree. Each node in the tree contains surface word form, word position, parent position, dependency type and POS tag. We use T to stand for our dependency tree, and $T(n)$ to stand for the subtree rooted at node n. Each subtree $T(n)$ covers a span of contiguous source words; for subspan $\bar{f}$ covered by $T(n)$, we say $\bar{f} \in T(n)$.

Cohesion is checked as we extend a state $(\bar{f}_h, HC_h)$ with the translation of $\bar{f}_{h+1}$, creating a new state $(\bar{f}_{h+1}, HC_{h+1})$. Algorithm 1 presents the cohesion check described by Cherry (2008). Line 2 selects focal points, based on the last translated phrase. Line 4 climbs from each focal point to find the largest subtree that needs to be completed before the translation process can move elsewhere in the tree. Line 5 checks each such subtree for completion. Since there are a constant number of focal points (always 2) and the tree climb and completion checks are both linear in the size of the source, the entire check can be shown to take linear time.

The selection of only two focal points is motivated by a "**violation free**" assumption. If one assumes that the

Proceedings of NAACL HLT 2009: Short Papers, pages 1–4,
Boulder, Colorado, June 2009.

Algorithm 1 Interruption Check (Coh1) (Cherry, 2008)

Input: Source tree T, previous phrase $\bar{f}_h$, current phrase $\bar{f}_{h+1}$, coverage vector HC

1: $Interruption \leftarrow False$
2: $F \leftarrow$ the left and right-most tokens of $\bar{f}_h$
3: **for** each of $f \in F$ **do**
4: Climb the dependency tree from f until you reach the highest node n such that $\bar{f}_{h+1} \notin T(n)$.
5: **if** n exists and $T(n)$ is not covered in HC_{h+1} **then**
6: $Interruption \leftarrow True$
7: **end if**
8: **end for**
9: Return $Interruption$

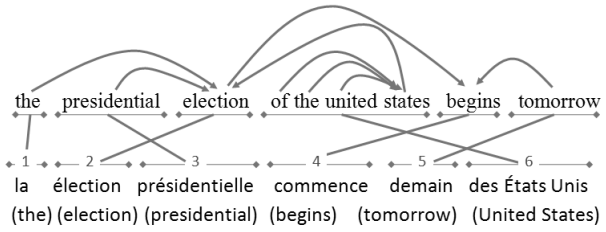


Figure 1: A candidate translation where Coh1 does not fire

translation represented by $(\bar{f}_h, HC_h)$ contains no cohesion violations, then checking only the end-points of $\bar{f}_h$ is sufficient to maintain cohesion. However, once a soft cohesion constraint has been implemented, this assumption no longer holds.

2 Extensions of Cohesive Constraints

2.1 Exhaustive Interruption Check (Coh2)

Because of the "violation free" assumption, Algorithm 1 implements the design decision to only suffer a violation penalty once, when cohesion is initially broken. However, this is not necessarily the best approach, as the decoder does not receive any further incentive to return to the partially translated subtree and complete it.

For example, Figure 1 illustrates a translation candidate of the English sentence "the presidential election of the united states begins tomorrow" into French. We consider $\bar{f}_4$ = "begins", $\bar{f}_5$ = "tomorrow". The decoder already translated "the presidential election" making the coverage vector HC_5 = "1 1 1 0 0 0 0 1 1". Algorithm 1 tells the decoder that no violation has been made by translating "tomorrow" while the decoder should be informed that there exists an outstanding violation. Algorithm 1 found the violation when the decoder previously jumped from "presidential" to "begins", and will not find another violation when it jumps from "begins" to "tomorrow".

Algorithm 2 is a modification of Algorithm 1, changing only line 2. The resulting system checks all previously covered tokens, instead of only the left and right-most tokens of $\bar{f}_{h+1}$, and therefore makes no violation-free assumption. For the example above, Algorithm 2 will inform the decoder that translating "tomorrow" also incurs a violation. Because $|F|$ is no longer constant, the time complexity of Coh2 is worse than Coh1. However, we can speed up the interruption check algorithm by hashing cohesion checks, so we only need to run Algorithm 2 once per $(\bar{f}_{h+1}, HC_{h+1})$.

Algorithm 2 Exhaustive Interruption Check (Coh2)

Input: Source tree T, previous phrase f_h, current phrase f_{h+1}, coverage vector HC

1: $Interruption \leftarrow False$
2: $F \leftarrow \{f | HC_h(f) = 1\}$
3: **for** each of $f \in F$ **do**
4: Climb the dependency tree from f until you reach the highest node n such that $\bar{f}_{h+1} \notin T(n)$.
5: **if** n exists and $T(n)$ is not covered in HC_{h+1} **then**
6: $Interruption \leftarrow True$
7: **end if**
8: **end for**
9: Return $Interruption$

Algorithm 3 Interruption Count (Coh3)

Input: Source tree T, previous phrase $\bar{f}_h$, current phrase $\bar{f}_{h+1}$, coverage vector HC

1: $ICount \leftarrow 0$
2: $F \leftarrow$ the left and right-most tokens of $\bar{f}_h$
3: **for** each of $f \in F$ **do**
4: Climb the dependency tree from f until you reach the highest node n such that $\bar{f}_{h+1} \notin T(n)$.
5: **if** n exists **then**
6: **for** each of $e \in T(n)$ and $HC_{h+1}(e) = 0$ **do**
7: $ICount = ICount + 1$
8: **end for**
9: **end if**
10: **end for**
11: Return $ICount$

2.2 Interruption Count (Coh3) and Exhaustive Interruption Count (Coh4)

Algorithm 1 and 2 described above interpret an interruption as a binary event. As it is possible to leave several words untranslated with a single jump, some interruptions may be worse than others. To implement this observation, an interruption count is used to assign a penalty to cohesion violations, based on the number of words left uncovered in the interrupted subtree. We initialize the interruption count with zero. At any search state when the cohesion violation is detected the count is incremented by

Algorithm 4 Exhaustive Interruption Count (Coh4)

Input: Source tree T, previous phrase f_h, current phrase f_{h+1}, coverage vector HC

1: $ICount \leftarrow 0$
2: $F \leftarrow \{f | HC_h(f) = 1\}$
3: **for** each of $f \in F$ **do**
4: Climb the dependency tree from f until you reach the highest node n such that $\bar{f}_{h+1} \notin T(n)$.
5: **if** n exists **then**
6: **for** each of $e \in T(n)$ and $HC_{h+1}(e) = 0$ **do**
7: $ICount = ICount + 1$
8: **end for**
9: **end if**
10: **end for**
11: Return $ICount$

one. The modification of Algorithm 1 and 2 lead to Interruption Count (Coh3) and Exhaustive Interruption Count (Coh4) algorithms, respectively. The changes only happen in lines 1, 5 and 6. We use an additional bit vector to make sure that if a node has been reached once during an interruption check, it should not be counted again. For the example in Section 2.1, Algorithm 4 will return 4 for $ICount$ ("of"; "the"; "united"; "states").

2.3 Rich Interruption Constraints (Coh5)

The cohesion constraints in Sections 2.1 and 2.2 do not leverage node information in the dependency tree structures. We propose the rich interruption constraints (Coh5) algorithm to combine four constraints which are Interruption, Interruption Count, Verb Count and Noun Count. The first two constraints are identical to what was described above. Verb and Noun count constraints are enforcing the following rule: a cohesion violation will be penalized more in terms of the number of verb and noun words that have not been covered. For example, we want to translate the English sentence "the presidential election of the united states begins tomorrow" to French with the dependency structure as in Figure 1. We consider $\bar{f}_h$ = "the united states", $\bar{f}_{h+1}$ = "begins". The coverage bit vector HC_{h+1} is "0 0 0 0 1 1 1 1 0". Algorithm 5 will return true for $Interruption$, 4 for $ICount$ ("the"; "presidential"; "election"; "of"), 0 for $VerbCount$ and 1 for $NounCount$ ("election").

Algorithm 5 Rich Interruption Constraints (Coh5)

Input: Source tree T, previous phrase $\bar{f}_h$, current phrase $\bar{f}_{h+1}$, coverage vector HC

1: $Interruption \leftarrow False$
2: $ICount, VerbCount, NounCount \leftarrow 0$
3: $F \leftarrow$ the left and right-most tokens of $\bar{f}_h$
4: **for** each of $f \in F$ **do**
5: Climb the dependency tree from f until you reach the highest node n such that $\bar{f}_{h+1} \notin T(n)$.
6: **if** n exists **then**
7: **for** each of $e \in T(n)$ and $HC_{h+1}(e) = 0$ **do**
8: $Interruption \leftarrow True$
9: $ICount = ICount + 1$
10: **if** POS of e is "VB" **then**
11: $VerbCount \leftarrow VerbCount + 1$
12: **else if** POS of e is "NN" **then**
13: $NounCount \leftarrow NounCount + 1$
14: **end if**
15: **end for**
16: **end if**
17: **end for**
18: Return $Interruption$, $ICount$, $VerbCount$, $NounCount$

3 Experiments

We built baseline systems using GIZA++ (Och and Ney, 2003), Moses' phrase extraction with grow-diag-final-end heuristic (Koehn et al., 2007), a standard phrase-based decoder (Vogel, 2003), the SRI LM toolkit (Stolcke, 2002), the suffix-array language model (Zhang and Vogel, 2005), a distance-based word reordering model with a window of 3, and the maximum number of target phrases restricted to 10. Results are reported using lowercase BLEU (Papineni et al., 2002). All model weights were trained on development sets via minimum-error rate training (MERT) (Och, 2003) with 200 unique n-best lists and optimizing toward BLEU. We used the MALT parser (Nivre et al., 2006) to obtain source English dependency trees and the Stanford parser for Arabic (Marneffe et al., 2006). In order to decide whether the translation output of one MT engine is significantly better than another one, we used the bootstrap method (Zhang et al., 2004) with 1000 samples ($p < 0.05$). We perform experiments on English→Iraqi and English→Spanish. Detailed corpus statistics are shown in Table 1. Table 2 shows results in lowercase BLEU and bold type is used to indicate highest scores. An italic text indicates the score is statistically significant better than the baseline.

	English→Iraqi		English→Spanish	
	English	Iraqi	English	Spanish
sentence pairs	654,556		1,310,127	
unique sent. pairs	510,314		1,287,016	
avg. sentence length	8.4	5.9	27.4	28.6
# words	5.5 M	3.8 M	35.8 M	37.4 M
vocabulary	34 K	109 K	117 K	173 K

Table 1: Corpus statistics

Our English-Iraqi data come from the DARPA TransTac program. We used TransTac T2T July 2007

	English→Iraqi		English→Spanish	
	july07	june08	ncd07	nct07
Baseline	31.58	23.58	33.18	32.04
+Coh1	**32.63**	24.45	33.49	*32.72*
+Coh2	32.51	**24.73**	33.52	*32.81*
+Coh3	32.43	24.19	33.37	*32.87*
+Coh4	32.32	24.66	33.47	*33.20*
+Coh5	31.98	24.42	**33.54**	***33.27***

Table 2: Scores of baseline and cohesion-enhanced systems on English→Iraqi and English→Spanish systems

(july07) as the development set and TransTac T2T June 2008 (june08) as the held-out evaluation set. Each test set has 4 reference translation. We applied the suffix-array LM up to 6-gram with Good-Turing smoothing. Our cohesion constraints produced improvements ranging between **0.5** and **1.2** BLEU point on the held-out evaluation set.

We used the Europarl and News-Commentary parallel corpora for English→Spanish as provided in the ACL-WMT 2008 shared task evaluation. The baseline system used the parallel corpus restricting sentence length to 100 words for word alignment and a 4-gram SRI LM with modified Kneyser-Ney smoothing. We used nc-devtest2007(ncd07) as the development set and nc-test2007(nct07) as the held-out evaluation set. Each test set has 1 translation reference. We obtained improvements ranging between **0.7** and **1.2** BLEU. All cohesion constraints perform statistically significant better than the baseline on the held-out evaluation set.

4 Conclusions

In this paper, we explored cohesive phrasal decoding, focusing on variants of cohesive constraints. We proposed four novel cohesive constraints namely exhaustive interruption check (Coh2), interruption count (Coh3), exhaustive interruption count (Coh4) and rich interruption constraints (Coh5). Our experimental results show that with cohesive constraints the system generates better translations in comparison with strong baselines. To ensure the robustness and effectiveness of the proposed approaches, we conducted experiments on 2 different language pairs, namely English→Iraqi and English→Spanish. These experiments also covered a wide range of training corpus sizes, ranging from 600K sentence pairs up to 1.3 million sentence pairs. All five proposed approaches give positive results. The improvements on English→Spanish are statistically significant at the 95% level. We observe a consistent pattern indicating that the improvements are stable in both language pairs.

Acknowledgments

This work is in part supported by the US DARPA TransTac programs. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA. We would like to thank Qin Gao and Matthias Eck for helpful conversations, Johan Hall and Joakim Nirve for helpful suggestions on training and using the English dependency model. We also thanks the anonymous reviewers for helpful comments.

References

Colin Cherry. 2008. Cohesive phrase-based decoding for statistical machine translation. In *Proceedings of ACL-08: HLT*, pages 72–80, Columbus, Ohio, June. Association for Computational Linguistics.

Heidi J. Fox. 2002. Phrasal cohesion and statistical machine translation. In *Proceedings of EMNLP'02*, pages 304–311, Philadelphia, PA, July 6-7.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of ACL'07*, pages 177–180, Prague, Czech Republic, June.

Marie-Catherine Marneffe, Bill MacCartney, and Christopher Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC'06*, Genoa, Italy.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2006. MaltParser: A data-driven parser-generator for dependency parsing. In *Proceedings of LREC'06*, Genoa, Italy.

Franz J. Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. In *Computational Linguistics*, volume 1:29, pages 19–51.

Franz Josef Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL'03*, pages 160–167.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL'02*, pages 311–318, Philadelphia, PA, July.

Andreas Stolcke. 2002. SRILM – An extensible language modeling toolkit. In *Proc. Intl. Conf. on Spoken Language Processing*, volume 2, pages 901–904, Denver.

Stephan Vogel. 2003. SMT decoder dissected: Word reordering. In *Proceedings of NLP-KE'03*, pages 561–566, Bejing, China, Oct.

Hirofumi Yamamoto, Hideo Okuma, and Eiichiro Sumita. 2008. Imposing constraints from the source tree on ITG constraints for SMT. In *Proceedings of the ACL-08: HLT, SSST-2*, pages 1–9, Columbus, Ohio, June. Association for Computational Linguistics.

Ying Zhang and Stephan Vogel. 2005. An efficient phrase-to-phrase alignment model for arbitrarily long phrase and large corpora. In *Proceedings of EAMT'05*, Budapest, Hungary, May. The European Association for Machine Translation.

Ying Zhang, Stephan Vogel, and Alex Waibel. 2004. Interpreting BLEU/NIST scores: How much improvement do we need to have a better system? In *Proceedings of LREC'04*, pages 2051–2054.

Revisiting Optimal Decoding for Machine Translation IBM Model 4

Sebastian Riedel[*†] **James Clarke**[‡]
[*]Department of Computer Science, University of Tokyo, Japan
[†]Database Center for Life Science, Research Organization of Information and System, Japan
[‡]Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, IL 61801
[*]sebastian.riedel@gmail.com [†]clarkeje@gmail.com

Abstract

This paper revisits optimal decoding for statistical machine translation using IBM Model 4. We show that exact/optimal inference using Integer Linear Programming is more practical than previously suggested when used in conjunction with the Cutting-Plane Algorithm. In our experiments we see that exact inference can provide a gain of up to one BLEU point for sentences of length up to 30 tokens.

1 Introduction

Statistical machine translation (MT) systems typically contain three essential components: (1) a model, specifying how the process of translation occurs; (2) learning regime, dictating the estimation of model's parameters; (3) decoding algorithm which provides the most likely translation of an input sentence given a model and its parameters.

The search space in statistical machine translation is vast which can make it computationally prohibitively to perform exact/optimal decoding (also known as search and MAP inference) especially since dynamic programming methods (such as the Viterbi algorithm) are typically not applicable. Thus greedy or heuristic beam-based methods have been prominent (Koehn et al., 2007) due to their efficiency. However, the efficiency of such methods have two drawbacks: (1) they are approximate and give no bounds as to how far their solution is away from the true optimum; (2) it can be difficult to incorporate additional generic global constraints into the search. The first point may be especially problematic from a research perspective as without bounds on the solutions it is difficult to determine whether the model or the search algorithm requires improvement for better translations.

Similar problems exist more widely throughout natural language processing where greedy based methods and heuristic beam search have been used in lieu of exact methods. However, recently there has been an increasing interest in using Integer Linear Programming (ILP) as a means to find MAP solutions. ILP overcomes the two drawbacks mentioned above as it is guaranteed to be exact, and has the ability to easily enforce global constraints through additional linear constraints. However, efficiency is usually sacrificed for these benefits.

Integer Linear Programming has previously been used to perform exact decoding for MT using IBM Model 4 and a bigram language model. Germann et al. (2004) view the translation process akin to the travelling salesman problem; however, from their reported results it is clear that using ILP naively for decoding does not scale up beyond short sentences (of eight tokens). This is due to the exponential number of constraints required to represent the decoding problem as an ILP program. However, work in dependency parsing (Riedel and Clarke, 2006) has demonstrated that it is possible to use ILP to perform efficient inference for very large programs when used in an incremental manner. This raises the question as to whether incremental (or Cutting-Plane) ILP can also be used to decode IBM Model 4 on real world sentences.

In this work we show that it is possible. Decoding IBM Model 4 (in combination with a bigram language model) using Cutting-Plane ILP scales to much longer sentences. This affords us the opportunity to finally analyse the performance of IBM Model 4 and the performance of its state-of-the-

Proceedings of NAACL HLT 2009: Short Papers, pages 5–8,
Boulder, Colorado, June 2009.

art ReWrite decoder. We show that using exact inference provides an increase of up to one BLEU point on two language pairs (French-English and German-English) in comparison to decoding using the ReWrite decoder. Thus the ReWrite decoder performs respectably but can be improved slightly, albeit at the cost of efficiency.

Although the community has generally moved away from word-based models, we believe that displaying optimal decoding in IBM Model 4 lays the foundations of future work. It is the first step in providing a method for researchers to gain greater insight into their translation models by mapping the decoding problem of other models into an ILP representation. ILP decoding will also allow the incorporation of global linguistic constraints in a manner similar to work in other areas of natural language processing.

The remainder of this paper is organised as follows: Sections 2 and 3 briefly recap IBM Model 4 and its ILP formulation. Section 4 reviews the Cutting-Plane Algorithm. Section 5 outlines our experiments and we end the paper with conclusions and a discussion of open questions for the community.

2 IBM Model 4

In this paper we focus on the translation model defined by IBM Model 4 (Brown et al., 1993). Translation using IBM Model 4 is performed by treating the translation process a noisy-channel model where the probability of the English sentence given a French sentence is, $P(\mathbf{e}|\mathbf{f}) = P(\mathbf{f}|\mathbf{e}) \cdot P(\mathbf{e})$, where $P(\mathbf{e})$ is a language model of English. IBM Model 4 defines $P(\mathbf{f}|\mathbf{e})$ and models the translation process as a generative process of how a sequence of target words (in our case French or German) is generated from a sequence of source words (English).

The generative story is as follows. Imagine we have an English sentence, $\mathbf{e} = e_1, \ldots, e_l$ and along with a NULL word (e_o) and French sentence, $\mathbf{f} = f_1, \ldots, f_m$. First a fertility is drawn for each English word (including the NULL symbol). Then, for each e_i we then independently draws a number of French words equal to e_i's fertility. Finally we process the English source tokens in sequence to determine the positions of their generated French target words. We refer the reader to Brown et al. (1993) for full details.

3 Integer Linear Programming Formulation

Given a trained IBM Model 4 and a French sentence $\mathbf{f}$ we need to find the English sentence $\mathbf{e}$ and alignment $\mathbf{a}$ with maximal $p(\mathbf{a}, \mathbf{e}|\mathbf{f}) \backsimeq p(\mathbf{e}) \cdot p(\mathbf{a}, \mathbf{f}|\mathbf{e})$.[1]

Germann et al. (2004) present an ILP formulation of this problem. In this section we will give a very high-level description of the formulation.[2] For brevity we refer the reader to the original work for more details.

In the formulation of Germann et al. (2004) an English translation is represented as the journey of a travelling salesman that visits one English token (hotel) per French token (city). Here the English token serves as the translation of the French one. A set of binary variables denote whether or not certain English token pairs are directly connected in this journey. A set of constraints guarantee that for each French token exactly one English token is visited. The formulation also contains an exponential number of constraints which forbid the possible cycles the variables can represent. It is this set of constraints that renders MT decoding with ILP difficult.

4 Cutting Plane Algorithm

The ILP program above has an exponential number of (cycle) constraints. Hence, simply passing the ILP to an off-the-shelf ILP solver is not practical for all but the smallest sentences. For this reason Germann et al. (2004) only consider sentences of up to eight tokens. However, recent work (Riedel and Clarke, 2006) has shown that even exponentially large decoding problems may be solved efficiently using ILP solvers if a Cutting-Plane Algorithm (Dantzig et al., 1954) is used.[3]

A Cutting-Plane Algorithm starts with a subset of the complete set of constraints. In our case this subset contains all but the (exponentially many) cycle constraints. The corresponding ILP is solved by a

[1] Note that in theory we should be maximizing $p(\mathbf{e}|\mathbf{f})$. However, this requires summation over all possible alignments and hence the problem is usually simplified as described here.

[2] Note that our actual formulation differs slightly from the original work because we use a first order modelling language that imposed certain restrictions on the type of constraints allowed.

[3] It is worth mentioning that Cutting Plane Algorithms have been successfully applied for solving very large instances of the Travelling Salesman Problem, a problem essentially equivalent to the decoding in IBM Model 4.

standard ILP solver, and the solution is inspected for cycles. If it contains no cycles, we have found the true optimum: the solution with highest score that does not violate any constraints. If the solution does contain cycles, the corresponding constraints are added to the ILP which is in turn solved again. This process is continued until no more cycles can be found.

5 Evaluation

In this section we describe our experimental setup and results.

5.1 Experimental setup

Our experimental setup is designed to answer several questions: (1) Is exact inference in IBM Model 4 possible for sentences of moderate length? (2) How fast is exact inference using Cutting-Plane ILP? (3) How well does the ReWrite Decoder[4] perform in terms of finding the optimal solution? (4) Does optimal decoding produce better translations?

In order to answer these questions we obtain a trained IBM Model 4 for French-English and German-English on Europarl v3 using GIZA++. A bigram language model with Witten-Bell smoothing was estimated from the corpus using the CMU-Cambridge Language Modeling Toolkit.

For exact decoding we use the two models to generate ILP programs for sentences of length up to (and including) 30 tokens for French and 25 tokens for German.[5] We filter translation candidates following Germann et al. (2004) by using only the top ten translations for each word[6] and a list of zero fertility words.[7] This resulted in 1101 French and 1062 German sentences for testing purposes. The ILP programs were then solved using the method described in Section 3. This was repeated using the ReWrite Decoder using the same models.

5.2 Results

The Cutting-Plane ILP decoder (which we will refer to as ILP decoder) produced output for 986 French sentences and 954 German sentences. From this we can conclude that it is possible to solve 90% of our sentences exactly using ILP. For the remaining 115 and 108 sentences we did not produce a solution due to: (1) the solver not completing within 30 minutes, or (2) the solver running out of memory.[8]

Table 1 shows a comparison of the results, broken down by input sentence length, obtained on the 986 French and 954 German sentences using the ILP and ReWrite decoders. First we turn our attention to the solve times obtained using ILP (for the sentences for which the solution was found within 30 minutes). The table shows that the average solve time is under one minute per sentence. As we increase the sentence length we see the solve time increases, however, we never see an order of magnitude increase between brackets as witnessed by Germann et al. (2004) thus optimal decoding is more practical than previously suggested. The average number of Cutting-Plane iterations required was 4.0 and 5.6 iterations for French and German respectively with longer sentences requiring more on average.

We next examine the performance of the two decoders. Following Germann et al. (2004) we define the ReWrite decoder as finding the optimal solution if the English sentence is the same as that produced by the ILP decoder. Table 1 shows that the ReWrite decoder finds the optimal solution 40.1% of the time for French and 29.1% for German. We also see the ReWrite decoder is less likely to find the optimal solution of longer sentences. We now look at the model scores more closely. The average log model error per token shows that the ReWrite decoder's error is proportional to sentence length and on average the ReWrite decoder is 2.2% away from the optimal solution in log space and 60.6% in probability space[9] for French, and 4.7% and 60.9% for German.

Performing exact decoding increases the BLEU score by 0.97 points on the French-English data set and 0.61 points on the German-English data set with similar performance increases observed for all sentence lengths.

6 Discussion and Conclusions

In this paper we have demonstrated that optimal decoding of IBM Model 4 is more practical than previously suggested. Our results and analysis show that

[4] Available at `http://www.isi.edu/licensed-sw/rewrite-decoder/`

[5] These limits were imposed to ensure the Python script generating the ILP programs did not run out of memory.

[6] Based on $t(e|f)$.

[7] Extracted using the rules in the filter script `rewrite.mkZeroFert.perl`

[8] All experiments were run on 3.0GHz Intel Core 2 Duo with 4GB RAM using a single core.

[9] These high error rates are an artefact of the extremely small probabilities involved.

Len	#	Solve Stats			BLEU		
		%Eq	Err	Time	ReW	ILP	Diff
1–5	21	85.7	15.0	0.7	56.5	56.2	-0.32
6–10	121	64.5	7.8	1.4	26.1	28.0	1.90
11–15	118	47.9	5.9	2.7	22.9	23.7	0.85
16–20	238	37.4	6.3	13.9	20.4	20.8	0.41
21–25	266	30.5	6.6	70.1	20.9	22.5	1.62
26–30	152	25.7	5.3	162.6	20.9	22.3	1.38
1–30	986	40.1	6.5	48.1	21.7	22.6	0.97

(a) French-English

Len	#	Solve Stats			BLEU		
		%Eq	Err	Time	ReW	ILP	Diff
1–5	31	83.9	27.4	0.8	40.7	41.1	0.44
6–10	175	51.4	19.7	1.7	19.2	20.9	1.72
11–15	242	30.6	17.4	5.5	16.0	16.7	0.72
16–20	257	19.1	14.4	23.9	15.8	15.9	0.16
21–25	249	15.7	14.0	173.4	15.3	15.9	0.61
1–25	954	29.1	16.4	53.5	16.1	16.7	0.61

(b) German-English

Table 1: Results on the two corpora. Len: range of sentence lengths; #: number of sentences in this range; %Eq: percentage of times ILP decoder returned same English sentence; Err: average difference between decoder scores per token ($\times 10^{-2}$) in log space; Time: the average solve time per sentence of ILP decoder in seconds; BLEU ReW, BLEU ILP, BLEU Diff: the BLEU scores of the output and difference between BLEU scores.

exact decoding has a practical purpose. It has allowed us to investigate and validate the performance of the ReWrite decoder through comparison of the outputs and model scores from the two decoders. Exact inference also provides an improvement in translation quality as measured by BLEU score.

During the course of this research we have encountered numerous challenges that were not apparent at the start. These challenges raise some interesting research questions and practical issues one must consider when embarking on exact inference using ILP. The first issue is that the generation of the ILP programs can take a long time. This leads us to wonder if there may be a way to provide tighter integration of program generation and solving. Such an integration would avoid the need to query the models in advance for *all* possible model components the solver may require.

Related to this issue is how to tackle the incorporation of higher order language models. Currently we use our bigram language model in a brute-force manner: in order to generate the ILP we evaluate the probability of all possible bigrams of English candidate tokens in advance. It seems clear that with higher order models this process will become prohibitively expensive. Moreover, even if the ILP could be generated efficiently, they will obviously be larger and harder to solve than our current ILPs. One possible solution may be the use of so-called delayed column generation strategies which incrementally add parts of the objective function (and hence the language model), but only when required by the ILP solver.[10]

[10]Note that delayed column generation is dual to performing cutting planes.

The use of ILP in other NLP tasks has provided a principled and declarative manner to incorporate global linguistic constraints on the system output. This work lays the foundations for incorporating similar global constraints for translation. We are currently investigating linguistic constraints for IBM Model 4 and other word-based models in general. A further extension is to reformulate higher-level MT models (phrase- and syntax-based) within the ILP framework. These representations could be more desirable from a linguistic constraint perspective as the formulation of constraints may be more intuitive.

Acknowledgements

We would like to thank Ulrich Germann and Daniel Marcu for their help with the ISI ReWrite Decoder.

References

Brown, Peter F., Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics* 19(2):263–311.

Dantzig, George B., Ray Fulkerson, and Selmer M. Johnson. 1954. Solution of a large-scale traveling salesman problem. *Operations Research* 2:393–410.

Germann, Ulrich, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2004. Fast and optimal decoding for machine translation. *Artificial Intelligence* 154(1-2):127–143.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondrej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: Open source toolkit for statistical machine translation. In *ACL 2009 Demos*. Prague, Czech Republic, pages 177–180.

Riedel, Sebastian and James Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *EMNLP 2006*. pages 129–137.

Efficient Extraction of Oracle-best Translations from Hypergraphs

Zhifei Li and **Sanjeev Khudanpur**
Center for Language and Speech Processing and Department of Computer Science
The Johns Hopkins University, Baltimore, MD 21218, USA
zhifei.work@gmail.com and khudanpur@jhu.edu

Abstract

Hypergraphs are used in several syntax-inspired methods of machine translation to compactly encode exponentially many translation hypotheses. The hypotheses closest to given *reference translations* therefore cannot be found via brute force, particularly for popular measures of closeness such as BLEU. We develop a dynamic program for extracting the so called *oracle-best hypothesis* from a hypergraph by viewing it as the problem of finding the most likely hypothesis under an n-gram *language model* trained from only the reference translations. We further identify and remove massive redundancies in the dynamic program state due to the sparsity of n-grams present in the reference translations, resulting in a very efficient program. We present runtime statistics for this program, and demonstrate successful application of the hypotheses thus found as the targets for discriminative training of translation system components.

1 Introduction

A *hypergraph*, as demonstrated by Huang and Chiang (2007), is a compact data-structure that can encode an exponential number of hypotheses generated by a regular phrase-based machine translation (MT) system (e.g., Koehn et al. (2003)) or a syntax-based MT system (e.g., Chiang (2007)). While the hypergraph represents a very large set of translations, it is quite possible that some desired translations (e.g., the reference translations) are not contained in the hypergraph, due to pruning or inherent deficiency of the translation model. In this case, one is often required to find the translation(s) in the hypergraph that are most similar to the desired translations, with similarity computed via some automatic metric such as BLEU (Papineni et al., 2002). Such maximally similar translations will be called *oracle-best* translations, and the process of extracting them *oracle extraction*. Oracle extraction is a nontrivial task because computing the similarity of any one hypothesis requires information scattered over many items in the hypergraph, and the exponentially large number of hypotheses makes a brute-force linear search intractable. Therefore, efficient algorithms that can exploit the structure of the hypergraph are required.

We present an efficient oracle extraction algorithm, which involves two key ideas. Firstly, we view the oracle extraction as a bottom-up *model scoring* process on a hypergraph, where the model is "trained" on the reference translation(s). This is similar to the algorithm proposed for a *lattice* by Dreyer et al. (2007). Their 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.

Efficient oracle extraction has at least three important applications in machine translation.

Discriminative Training: In discriminative training, the objective is to tune the model parameters, e.g. weights of a perceptron model or conditional random field, such that the reference translations are preferred over competitors. However, the reference translations may not be reachable by the translation system, in which case the oracle-best hypotheses should be substituted in training.

Proceedings of NAACL HLT 2009: Short Papers, pages 9–12,
Boulder, Colorado, June 2009.

System Combination: In a typical *system combination* task, e.g. Rosti et al. (2007), each component system produces a set of translations, which are then grafted to form a confusion network. The confusion network is then rescored, often employing additional (language) models, to select the final translation. When measuring the goodness of a hypothesis in the confusion network, one requires its score under each component system. However, some translations in the confusion network may not be reachable by some component systems, in which case a system's score for the most similar reachable translation serves as a good approximation.

Multi-source Translation: In a multi-source translation task (Och and Ney, 2001) the input is given in *multiple* source languages. This leads to a situation analogous to system combination, except that each component translation system now corresponds to a specific source language.

2 Oracle Extraction on a Hypergraph

In this section, we present the oracle extraction algorithm: it extracts one or more translations in a hypergraph that have the maximum BLEU score[1] with respect to the corresponding reference translation(s).

The BLEU score of a hypothesis h relative to a reference r may be expressed in the log domain as,

$$\log \text{BLEU}(r,h) = \min\left[1-\frac{|r|}{|h|},0\right] + \sum_{n=1}^{4}\frac{1}{4}\log p_n.$$

The first component is the brevity penalty when $|h|<|r|$, while the second component corresponds to the geometric mean of n-gram precisions p_n (with clipping). While BLEU is normally defined at the corpus level, we use the sentence-level BLEU for the purpose of oracle extraction.

Two key ideas for extracting the oracle-best hypothesis from a hypergraph are presented next.

2.1 Oracle Extraction as Model Scoring

Our *first* key idea is to view the oracle extraction as a bottom-up *model scoring* process on the hypergraph. Specifically, we train a 4-gram language model (LM) on only the *reference translation(s)*, and use this LM as the *only* model to do a Viterbi search on the hypergraph to find the hypothesis that has the maximum (oracle) LM score. Essentially, the LM is simply a table memorizing the counts of n-grams found in the reference translation(s), and the LM score is the log-BLEU value (instead of log-probability, as in a regular LM). During the search, the dynamic programming (DP) states maintained at each item include the left- and right-side LM context, and the length of the partial translation. To compute the n-gram precisions p_n incrementally during the search, the algorithm also memorizes at each item a vector of maximum numbers of n-gram matches between the partial translation and the reference(s). Note however that the *oracle state* of an item (which decides the uniqueness of an item) depends only on the LM contexts and span lengths, not on this vector of n-gram match counts.

The computation of BLEU also requires the brevity penalty, but since there is no explicit alignment between the source and the reference(s), we cannot get the exact reference length $|r|$ at an intermediate item. The exact value of brevity penalty is thus not computable. We approximate the true reference length for an item with a *product* between the *length* of the source string spanned by that item and a *ratio* (which is between the lengths of the whole reference and the whole source sentence). Another approximation is that we do not consider the effect of clipping, since it is a global feature, making the strict computation intractable. This does not significantly affect the quality of the oracle-best hypothesis as shown later. Table 1 shows an example how the BLEU scores are computed in the hypergraph.

The process above may be used either in a first-stage decoding or a hypergraph-rescoring stage. In the latter case, if the hypergraph generated by the first-stage decoding does not have a set of DP states that is a superset of the DP states required for oracle extraction, we need to split the items of the first-stage hypergraph and create new items with sufficiently detailed states.

It is worth mentioning that if the hypergraph items contain the state information necessary for extracting the oracle-best hypothesis, it is straightforward to further extract the k-best hypotheses in the hypergraph (according to BLEU) for any $k \geq 1$ using the algorithm of Huang and Chiang (2005).

[1] We believe our method is general and can be extended to other metrics capturing only n-gram dependency and other compact data structures, e.g. lattices.

Item	$\|\mathbf{h}\|$	$\|\widetilde{\mathbf{r}}\|$	matches	$\log$ **BLEU**
Item A	5	6.2	(3, 2, 2, 1)	-0.82
Item B	10	9.8	(8, 7, 6, 5)	-0.27
Item C	17	18.3	(12, 10, 9, 6)	-0.62

Table 1: Example computation when items `A` and `B` are combined by a rule to produce item `C`. $|\widetilde{\mathbf{r}}|$ is the approximated reference length as described in the text.

2.2 Equivalent Oracle State Maintenance

The process above, while able to extract the oracle-best hypothesis from a hypergraph, is very slow due to the need to maintain a dedicated item for each *oracle state* (i.e., a combination of left-LM state, right-LM state, and hypothesis length). This is especially true if the baseline system uses a LM whose order is smaller than four, since we need to split the items in the original hypergraph into many sub-items during the search. To speed up the extraction, our *second* key idea is to maintain an *equivalent oracle state*.

Roughly speaking, instead of maintaining a different state for different language model words, we collapse them into a single state whenever it does not affect BLEU. For example, if we have two left-side LM states `a b c` and `a b d`, and we know that the reference(s) do not have any n-gram ending with them, then we can reduce them both to `a b` and ignore the last word. This is because the combination of neither left-side LM state (`a b c` or `a b d`) can contribute an n-gram match to the BLEU computation, regardless of which prefix in the hypergraph they combine with. Similarly, if we have two right-side LM states `a b c` and `d b c`, and if we know that the reference(s) do not have any n-gram starting with either, then we can ignore the first word and reduce them both to `b c`. We can continue this reduction recursively as shown in Figures 1 and 2, where IS-A-PREFIX(e_i^m) (or IS-A-SUFFIX(e_1^i)) checks if e_i^m (resp. e_1^i) is a prefix (suffix) of any n-gram in the reference translation(s). For BLEU, $1 \leq n \leq 4$.

This *equivalent oracle state maintenance* technique, in practice, dramatically reduces the number of distinct items preserved in the hypergraph for oracle extraction. To understand this, observe that if all hypotheses in the hypergraph together contain m unique n-grams, for any fixed n, then the total number of equivalent items takes a multiplicative factor that is $O(m^2)$ due to left- and right-side LM state

EQ-L-STATE (e_1^m)

```
els ← e_1^m
for i ← m to 1                    ▷ right to left
    if IS-A-SUFFIX(e_1^i)
        break                     ▷ stop reducing els
    else
        els ← e_1^{i-1}           ▷ reduce state
return els
```

Figure 1: Equivalent Left LM State Computation.

EQ-R-STATE (e_1^m)

```
ers ← e_1^m
for i ← 1 to m                    ▷ left to right
    if IS-A-PREFIX (e_i^m)
        break                     ▷ stop reducing ers
    else
        ers ← e_{i+1}^m           ▷ reduce state
return ers
```

Figure 2: Equivalent Right LM State Computation.

maintenance of Section 2.1. This multiplicative factor under the equivalent state maintenance above is $O(\tilde{m}^2)$, where $\tilde{m}$ is the number of unique n-grams in the reference translations. Clearly, $\tilde{m} \ll m$ by several orders of magnitude, leading to effectively much fewer items to process in the chart.

One may view this idea of maintaining equivalent states more generally as an *outside* look-ahead during bottom-up *inside* parsing. The look-ahead uses some external information, e.g. IS-A-SUFFIX$(\cdot)$, to *anticipate* whether maintaining a detailed state now will be of consequence later; if not then the inside parsing eliminates or collapses the state into a coarser state. The technique proposed by Li and Khudanpur (2008a) for decoding with large LMs is a special case of this general theme.

3 Experimental Results

We report experimental results on a Chinese to English task, for a system that is trained using a similar pipeline and data resource as in Chiang (2007).

3.1 Goodness of the Oracle-Best Translations

Table 2 reports the average speed (seconds/sentence) for oracle extraction. Hypergraphs were generated with a trigram LM and expanded on the fly for 4-gram BLEU computation.

Basic DP	Collapse equiv. states	speed-up
25.4 sec/sent	0.6 sec/sent	$\times$ 42

Table 2: Speed of oracle extraction from hypergraphs. The basic dynamic program (Sec. 2.1) improves significantly by collapsing *equivalent oracle states* (Sec. 2.2).

Table 3 reports the goodness of the oracle-best hypotheses on three standard data sets. The highest achievable BLEU score in a hypergraph is clearly much higher than in the 500-best *unique* strings. This shows that a hypergraph provides a much better basis, e.g., for reranking than an n-best list.

As mentioned in Section 2.1, we use several approximations in computing BLEU (e.g., no clipping and approximate reference length). To justify these approximations, we first extract 500-best unique *oracles* from the hypergraph, and then rerank the oracles based on the true sentence-level BLEU. The last row of Table 3 reports the reranked one-best oracle BLEU scores. Clearly, the approximations do not hurt the oracle BLEU very much.

Hypothesis space	MT'04	MT'05	MT'06
1-best (Baseline)	35.7	32.6	28.3
500-unique-best	44.0	41.2	35.1
Hypergraph	52.8	51.8	37.8
500-best oracles	**53.2**	**52.2**	**38.0**

Table 3: Baseline and oracle-best 4-gram BLEU scores with 4 references for NIST Chinese-English MT datasets.

3.2 Discriminative Hypergraph-Reranking

Oracle extraction is a critical component for hypergraph-based discriminative reranking, where millions of model parameters are discriminatively tuned to prefer the oracle-best hypotheses over others. Hypergraph-reranking in MT is similar to the forest-reranking for *monolingual parsing* (Huang, 2008). Moreover, once the oracle-best hypothesis is identified, discriminative models may be trained on *hypergraphs* in the same way as on n-*best* lists (cf e.g. Li and Khudanpur (2008b)). The results in Table 4 demonstrate that hypergraph-reranking with a discriminative LM or TM improves upon the baseline models on all three test sets. Jointly training both the LM and TM likely suffers from over-fitting.

Test Set	MT'04	MT'05	MT'06
Baseline	35.7	32.6	28.3
Discrim. LM	35.9	33.0	28.2
Discrim. TM	**36.1**	**33.2**	**28.7**
Discrim. TM+LM	36.0	33.1	28.6

Table 4: BLEU scores after discriminative hypergraph-reranking. Only the language model (LM) or the translation model (TM) or both (LM+TM) may be discriminatively trained to prefer the oracle-best hypotheses.

4 Conclusions

We have presented an efficient algorithm to extract the oracle-best translation hypothesis from a hypergraph. To this end, we introduced a novel technique for *equivalent oracle state maintenance*, which significantly speeds up the oracle extraction process. Our algorithm has clear applications in diverse tasks such as discriminative training, system combination and multi-source translation.

References

D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201-228.

M. Dreyer, K. Hall, and S. Khudanpur. 2007. Comparing Reordering Constraints for SMT Using Efficient BLEU Oracle Computation. *In Proc. of SSST.*

L. Huang. 2008. Forest Reranking: Discriminative Parsing with Non-Local Features. *In Proc. of ACL.*

L. Huang and D. Chiang. 2005. Better k-best parsing. *In Proc. of IWPT*.

L. Huang and D. Chiang. 2007. Forest Rescoring: Faster Decoding with Integrated Language Models. *In Proc. of ACL.*

P. Koehn, F. J. Och, and D. Marcu.2003. Statistical phrase-based translation. *In Proc. of NAACL.*

Z. Li and S. Khudanpur. 2008a. A Scalable Decoder for Parsing-based Machine Translation with Equivalent Language Model State Maintenance. *In Proc. SSST*.

Z. Li and S. Khudanpur. 2008b. Large-scale Discriminative n-gram Language Models for Statistical Machine Translation. *In Proc. of AMTA*.

F. Och and H. Ney. 2001. Statistical multisource translation. *In Proc. MT Summit VIII*.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. *In Proc. of ACL.*

A.I. Rosti, S. Matsoukas, and R. Schwartz. 2007. Improved word-level system combination for machine translation. *In Proc. of ACL.*

Semantic Roles for SMT: A Hybrid Two-Pass Model

Dekai WU[1] **Pascale FUNG**[2]

Human Language Technology Center
HKUST
[1]Department of Computer Science and Engineering
[2]Department of Electronic and Computer Engineering
University of Science and Technology, Clear Water Bay, Hong Kong
dekai@cs.ust.hk pascale@ee.ust.hk

Abstract

We present results on a novel hybrid semantic SMT model that incorporates the strengths of both semantic role labeling and phrase-based statistical machine translation. The approach avoids major complexity limitations via a two-pass architecture. The first pass is performed using a conventional phrase-based SMT model. The second pass is performed by a re-ordering strategy guided by shallow semantic parsers that produce both semantic frame and role labels. Evaluation on a Wall Street Journal newswire genre test set showed the hybrid model to yield an improvement of roughly half a point in BLEU score over a strong pure phrase-based SMT baseline – to our knowledge, the first successful application of semantic role labeling to SMT.

1 Introduction

Many of the most glaring errors made by today's statistical machine translation systems are those resulting from confusion of semantic roles. Translation errors of this type frequently result in critical misunderstandings of the essential meaning of the original input language sentences – who did what to whom, for whom or what, how, where, when, and why.

Semantic role confusions are errors of adequacy rather than fluency. It has often been noted that the dominance of lexically-oriented, precision-based metrics such as BLEU (Papineni *et al.* 2002) tend to reward fluency more than adequacy. The length penalty in the BLEU metric, in particular, is only an indirect and weak indicator of adequacy. As a result, SMT work has been driven to optimize systems such that they often produce translations that contain significant role confusion errors despite reading fluently.

The present work is inspired by the question of whether we can improve translation utility via a strategy of favoring semantic adequacy slightly more – possibly at the expense of slight degradations in lexical fluency.

Shallow semantic parsing models have attained increasing levels of accuracy in recent years (Gildea and Jurafsky 2000; Sun and Jurafsky 2004; Pradhan *et al.* 2004, 2005; Pradhan 2005; Fung *et al.* 2006, 2007; Giménez and Màrquez 2007a, 2008). Such models, which identify semantic frames within input sentences by marking its predicates, and labeling their arguments with the semantic roles that they fill.

Evidence has begun to accumulate that semantic frames – predicates and semantic roles – tend to preserve consistency across translations better than syntactic roles do. This is, of course, by design; it follows from the definition of semantic roles, which are less language-dependent than syntactic roles. Across Chinese and English, for example, it has been reported that approximately 84% of semantic roles are preserved consistently (Fung *et al.* 2006). Of these, roughly 15% do *not* preserve syntactic roles consistently.

Since this directly targets the task of determining semantic correctness, we believe that the adequacy of MT output could be improved by leveraging the predictions of semantic parsers. We would like to exploit automatic semantic parsers to identify inconsistent semantic frame and role mappings between the input source sentences and their output translations.

However, we take note of the difficult experience in making syntactic and semantic models con-

Proceedings of NAACL HLT 2009: Short Papers, pages 13–16,
Boulder, Colorado, June 2009.

tribute to improving SMT accuracy. On the one hand, there is reason to be optimistic. Over the past decade, we have seen an accumulation of evidence that SMT accuracy can be improved via tree-structured and syntactic models (e.g., Wu 1997; Wu and Chiang 2009), and more recently, work from lexical semantics has also at long last been successfully applied to increasing SMT accuracy, in the form of techniques adapted from word sense disambiguation models (Chan *et al.* 2007; Giménez and Màrquez 2007b; Carpuat and Wu 2007). On the other hand, both directions saw unexpected disappointments along the way (e.g., Och *et al.* 2003; Carpuat and Wu 2005). We are therefore forewarned that it is likely to be at least as difficult to successfully adapt the even more complex types of lexical semantics modeling from semantic parsing and role labeling to the translation task.

In this paper, we present a novel hybrid model that, for the first time to our knowledge, successfully applies semantic parsing technology to the challenge of improving the quality of Chinese-English statistical machine translation. The model makes use of a typical representative SMT system based on Moses, plus shallow semantic parsers for both English and Chinese.

2 Hybrid two-pass semantic SMT

While the accuracy of shallow semantic parsers has been approaching reasonably high levels in recent years for well-studied languages like English, and to a lesser extent, Chinese, the problem of excessive computational complexity is one of the primary challenges in adapting semantic parsing technology to the translation task.

Semantic parses, by definition, are less likely than syntactic parses to obey clearly nested hierarchical composition rules. Moreover, the semantic parses are less likely to share an exactly isomorphic structure across the input and output languages, since the raison d'être of semantic parsing is to capture semantic frame and role regularities independent of syntactic variation – monolingually and cross-lingually.

This makes it difficult to incorporate semantic parsing into SMT merely by applying the sort of dynamic programming techniques found in current syntactic and tree-structured SMT models, most of which rely on being able to factor the computation into independent computations on the subtrees. In other words, the key computational obstacle is that the semantic parse of a larger string (or string pair, in the case of translation) is not in general strictly mechanically composable from the semantic parses of its smaller substrings (or substring pairs).

In fact, the lack of easy compositionality is the reason that today's most accurate shallow semantic parsers rely not primarily on compositional parsing techniques, but rather on ensembles of predictors that independently rate/rank a wide variety of factors supporting the role assignments given a broad *sentence-wide* range of context features. But while this improves semantic parsing accuracy, it poses a major obstacle for efficient tight integration into the sub-hypothesis construction and maintenance loops within SMT decoders.

To circumvent this computational obstacle, the hybrid two-pass model defers application of the non-compositional semantic parsing information until a second error-correcting pass. This imposes a division of labor between the two passes.

---

1. Apply a semantic parser for the *input* language to the input source sentence.
2. Apply a semantic parser for the *output* language to the baseline translation that was output by the first pass. Note: this also produces a shallow syntactic parse as a byproduct.
3. If the semantic frames (target predicates and their associated semantic roles) are all consistent between the input and output sentences, and are aligned to each other by the phrase alignments from the first pass, then finish immediately and output the baseline translation.
4. Segment the baseline translation by introducing segment boundaries around every constituent phrase whose shallow syntactic parse category (from step 2) was V, NP, or PP. This breaks the baseline translation into a small number of coarse chunks to consider during re-ordering, instead of a large number of individual words.
5. Generate a set of candidate re-ordered translation hypotheses by iteratively moving constituent phrases whose predicate or semantic role label was *mismatched* to the input sentence. Each new candidate generated may in turn spawn a further set of candidates (especially since moving one constituent phrase may cause another's predicate or semantic role label to change from matched to mismatched). This search is performed breadth-first to favor fewer re-orderings (in case the hypothesis generation grows beyond allotted time).
6. Apply a semantic parser for the *output* language to each candidate re-ordered translation hypothesis as it is generated.
7. Return the re-ordered translation hypothesis with the maximum match of semantic predicates and arguments.

Figure 1. Algorithm for second pass.

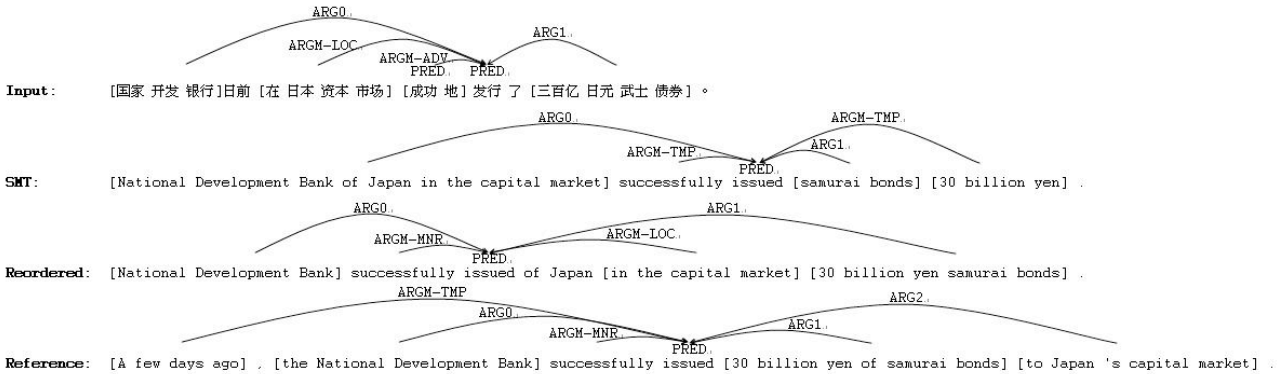


Figure 2. Example, showing translations after SMT first pass and after re-ordering second pass.

The first pass is performed using a conventional phrase-based SMT model. The phrase-based SMT model is assigned to the tasks of (a) providing an initial baseline hypothesis translation, and (b) fixing the lexical choice decisions. Note that the lexical choice decisions are *not* only at the single-word level, but are in general at the *phrasal* level.

The second pass takes the output of the first pass, and re-orders constituent phrases corresponding to semantic predicates and arguments, seeking to maximize the cross-lingual match of the semantic parse of the re-ordered translation to that of the original input sentence. The second pass algorithm performs the error correction shown in Figure 1.

The design decision to allow the first pass to fix all lexical choices follows an insight inspired by an empirical observation from our error analyses: the lexical choice decisions being made by today's SMT models have attained fairly reasonable levels, and are not where the major problems of adequacy lie. Rather, the ordering of arguments in relation to their predicates is often where the main failures of adequacy occur. By avoiding lexical choice variations while considering re-ordering hypotheses, a significantly larger amount of re-ordering can be done without further increasing computational complexity. So we sacrifice a small amount of fluency by allowing re-ordering without compensating lexical choice – in exchange for gaining potentially a larger amount of fluency by getting the predicate-argument structure right.

The model has a similar rationale for employing a re-ordering pass instead of re-ranking *n*-best lists or lattices. Oracle analysis of *n*-best lists and lattices show that they often focus on lexical choice alternatives rather than re-ordering / role variations which are more important to semantic adequacy.

3 Experiment

A Chinese-English experiment was conducted on the two-pass hybrid model. A phrase-based SMT baseline model was built by augmenting the open source statistical machine translation decoder Moses (Koehn *et al.* 2007) with additional pre-processors. English and Chinese shallow semantic parsers followed those discussed in Section 1.

The model was trained on LDC newswire parallel text consisting of 3.42 million sentence pairs, containing 64.1 million English words and 56.9 million Chinese words. The English was tokenized and case-normalized; the Chinese was tokenized via a maximum-entropy model (Fung *et al.* 2004).

Phrase translations were extracted via the grow-diag-final heuristic.

The language model is a 6-gram model trained with Kneser-Ney smoothing using the SRI language modeling toolkit (Stolcke 2002).

The test set of Wall Street Journal newswire sentences was randomly extracted from the Chinese-English Bilingual Propbank. Although we did not make use of the Propbank annotations, this would potentially allow other types of analyses in the future.

The phrase-based SMT model used for the first pass achieves a BLEU score of 42.99, establishing a fairly strong baseline to begin with.

In comparison, the automatically error-corrected translations that are output by the second pass achieve a BLEU score of 43.51. This represents approximately half a point improvement over the strong baseline.

An example is seen in Figure 2. The SMT first pass translation has an ARG0 *National Development Bank of Japan in the capital market* which is badly mismatched to *both* the input sentence's

ARG0 国家 开发 银行 and ARGM-LOC 在 日本 资本 市场. The second pass ends up re-ordering the constituent phrase corresponding to the mismatched ARGM-LOC, *of Japan in the capital market*, to follow the PRED *issued*, where the new English semantic parse now assigns most of its words the correctly matched ARGM-LOC semantic role label. Similarly, *samurai bonds 30 billion yen* is re-ordered to *30 billion yen samurai bonds*.

4 Discussion and conclusion

To our knowledge, this is a first result demonstrating that shallow semantic parsing can improve translation accuracy of SMT models. We note that accuracy here was measured via BLEU, and it has been widely observed that the negative impacts of semantic predicate-argument errors on the utility of the translation are underestimated by evaluation metrics based on lexical criteria such as BLEU. We conjecture that more expensive manual evaluation techniques which directly measure translation utility could even more strongly reveal improvement in role confusion errors.

The hybrid two-pass approach can be compared with the greedy re-ordering based strategy of the ReWrite decoder (Germann *et al.* 2001), although our search is breadth-first rather than purely greedy. Whereas ReWrite was based on word-level re-ordering, however, our approach is based on constituent phrase re-ordering, and the phrases to be re-ordered are more selectively chosen via the semantic parse labels. Moreover, the objective function being maximized by ReWrite is still the SMT model score; whereas in our case the new objective function is cross-lingual semantic predicate-argument match (plus an implicit search bias toward fewer re-orderings).

The hybrid two-pass approach can also be compared with serial combination architectures for hybrid MT (e.g., Ueffing *et al.* 2008). But whereas Ueffing *et al.* take the output from a first-pass rule-based MT system, and then correct it using a second-pass SMT system, our two-pass semantic SMT model does the reverse: it takes the output from a first-pass SMT system, and then corrects it with the aid of semantic analyzers.

Acknowledgments. Thanks to Chi-kiu Lo and Zhaojun Wu. This material is based upon work supported in part by the Defense Advanced Research Projects Agency (DARPA) under GALE Contract No. HR0011-06-C-0023, and by the Hong Kong Research Grants Council (RGC) research grants GRF621008, GRF612806, DAG03/04.EG09, RGC6256/00E, and RGC6083/99E.

References

Marine Carpuat and Dekai Wu. 2005. Word sense disambiguation vs. statistical machine translation. *43rd Annual Meeting of the Association for Computational Linguistics (ACL-2005)*. Ann Arbor, MI: Jun 2005.

Marine Carpuat and Dekai Wu. 2007. Improving statistical machine translation using word sense disambiguation. *2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2007)*. Prague: Jun 2007. 61-72.

Yee Seng Chan, Hwee Tou Ng and David Chiang 2007. Word sense disambiguation improves statistical machine translation. *45th Annual Meeting of the Association for Computational Linguistics (ACL-07)*, Prague: Jun 2007.

Pascale Fung, Grace Ngai, Yongsheng Yang and Benfeng Chen. 2004. A maximum-entropy Chinese parser augmented by transformation-based learning. *ACM Transactions on Asian Language Information Processing (TALIP)* **3**(2): 159-168.

Pascale Fung, Zhaojun Wu, Yongsheng Yang and Dekai Wu. 2006. Automatic learning of Chinese/English semantic structure mapping. *IEEE/ACL 2006 Workshop on Spoken Language Technology (SLT 2006)*. Aruba: Dec 2006. 230-233.

Pascale Fung, Zhaojun Wu, Yongsheng Yang and Dekai Wu. 2007. Learning Bilingual Semantic Frames: Shallow Semantic Parsing vs. Semantic Role Projection. *11th Conference on Theoretical and Methodological Issues in Machine Translation (TMI 2007)*. Skövde, Sweden: Sep 2007. 75-84.

Ulrich Germann, Michael Jahr, Kevin Knight, Daniel Marcu and Kenji Yamada. Fast Decoding and Optimal Decoding for Machine Translation. 2001. *39th Annual Meeting of the Association for Computational Linguistics* (ACL-2001). Toulouse: July 2001.

Daniel Gildea and Daniel Jurafsky. 2000. Automatic Labeling of Semantic Roles. *38th Annual Conference of the Association for Computational Linguistics (ACL-2000)*. 512–520, Hong Kong: Oct 2000.

Jesús Giménez and Lluís Màrquez. 2007. Linguistic Features for Automatic Evaluation of Heterogeneous MT Systems. *WMT 2007 (ACL'07)*.

Jesús Giménez and Lluís Màrquez. 2007. Discriminative Phrase Selection for Statistical Machine Translation. *Learning Machine Translation*. NIPS Workshop Series. MIT Press.

Jesús Giménez and Lluís Màrquez. 2008. A Smorgasbord of Features for Automatic MT Evaluation. *3rd ACL Workshop on Statistical Machine Translation (shared evaluation task)*. Pages 195-198, Columbus, Ohio: Jun 2008.

Alessandro Moschitti and Roberto Basili. 2005. Verb subcategorization kernels for automatic semantic labeling. *ACL-SIGLEX Workshop on Deep Lexical Acquisition*. Ann Arbor: Jun 2005. 10-17.

Franz Josef Och, Daniel Gildea, Sanjeev Khudanpur, Anoop Sarkar, Kenji Yamada, Alex Fraser, Shankar Kumar, Libin Shen, David Smith, Katherine Eng, Viren Jain, Zhen Jin and Dragomir Radev. 2004. A smorgasbord of features for statistical machine translation. *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL-2004)*. Boston: May 2004.

Kishore Papineni, Salim Roukos, Todd Ward and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. *40th Annual Meeting of the Association for Computational Linguistics* (ACL-2002).

Sameer Pradhan. 2005. *ASSERT: Automatic Statistical SEmantic Role Tagger*. http://oak.colorado.edu/assert/.

Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H. Martin and Daniel Jurafsky. 2005. Support Vector Learning for Semantic Argument Classification. *Machine Learning* **60**(1-3): 11-39.

Sameer Pradhan, Wayne Ward, Kadri Hacioglu, James Martin and Daniel Jurafsky. 2004. Shallow Semantic Parsing using Support Vector Machines. *Human Language Technology/North American Chapter of the Association for Computational Linguistics (HLT/NAACL-2004)*. Boston: May 2004.

Andreas Stolcke. 2002. SRILM – An Extensible Language Modeling Toolkit. International Conference on Spoken Language Processing (ICSLP-2002). Denver, Colorado: Sep 2002.

Honglin Sun and Daniel Jurafsky. 2004. Shallow Semantic Parsing of Chinese. *Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL-2004)*. 249-256. Boston: May 2004.

Nicola Ueffing, Jens Stephan, Evgeny Matusov, Loïc Dugast, George Foster, Roland Kuhn, Jean Senellart and Jin Yang. 2008. Tighter Integration of Rule-based and Statistical MT in Serial System Combination. *22nd International Conference on Computational Linguistics (COLING 2008)*. Manchester: Aug 2008.

Dekai Wu. 1997. Stochastic Inversion Transduction Grammars and bilingual parsing of parallel corpora. *Computational Linguistics* **23**(3): 377-404.

Dekai Wu and David Chiang (eds). 2009. *Proceedings of SSST-3, Third Workshop on Syntax and Structure in Statistical Translation* (NAACL-HLT 2009). Boulder, CO: Jun 2009.

Nianwen Xue and Martha Palmer. 2005. Automatic Semantic Role Labeling for Chinese Verbs. *19th International Joint Conference on Artificial Intelligence*. Edinburgh, Scotland.

Comparison of Extended Lexicon Models in Search and Rescoring for SMT

Saša Hasan and Hermann Ney
Human Language Technology and Pattern Recognition Group
Chair of Computer Science 6, RWTH Aachen University, Germany
{hasan,ney}@cs.rwth-aachen.de

Abstract

We show how the integration of an extended lexicon model into the decoder can improve translation performance. The model is based on lexical triggers that capture long-distance dependencies on the sentence level. The results are compared to variants of the model that are applied in reranking of n-best lists. We present how a combined application of these models in search and rescoring gives promising results. Experiments are reported on the GALE Chinese-English task with improvements of up to +0.9% BLEU and -1.5% TER absolute on a competitive baseline.

1 Introduction

Phrase-based statistical machine translation has improved significantly over the last decade. The availability of large amounts of parallel data and access to open-source software allow for easy setup of translation systems with acceptable performance. Public evaluations such as the NIST MT Eval or the WMT Shared Task help to measure overall progress within the community. Most of the groups use a phrase-based decoder (e.g. Pharaoh or the more recent Moses) based on a log-linear fusion of models that enable the avid researcher to quickly incorporate additional features and investigate the effect of additional knowledge sources to guide the search for better translation hypotheses.

In this paper, we deal with an extended lexicon model and its incorporation into a state-of-the-art decoder. We compare the results of the integration to a similar setup used within a rescoring framework and show the benefits of integrating additional models directly into the search process. As will be shown, although a rescoring framework is suitable for obtaining quick trends of incorporating additional models into a system, an alternative that includes the model in search should be preferred. The integration does not only yield better performance, we will also show the benefit of combining both approaches in order to boost translation quality even more. The extended lexicon model which we apply is motivated by a trigger-based approach (Hasan et al., 2008). A standard lexicon modeling dependencies of target and source words, i.e. $p(e|f)$, is extended with a second trigger f' on the source side, resulting in $p(e|f, f')$. This model allows for a more fine-grained lexical choice of the target word depending on the additional source word f'. Since the second trigger can move over the whole sentence, we capture global (sentence-level) context that is not modeled in local n-grams of the language model or in bilingual phrase pairs that cover only a limited amount of consecutive words.

Related work A similar approach has been tried in the word-sense disambiguation (WSD) domain where local but also across-sentence unigram collocations of words are used to refine phrase pair selection dynamically by incorporating scores from the WSD classifier (Chan et al., 2007). A maximum-entropy based approach with different features of surrounding words that are locally bound to a context of three positions to the left and right is reported in (García-Varea et al., 2001). A logistic regression-based word translation model is investigated by Vickrey et al. (2005) but has not been evaluated on a machine translation task. Another WSD approach incorporating context-dependent phrasal translation lexicons is presented by Carpuat and Wu (2007) and has been evaluated on several translation

Proceedings of NAACL HLT 2009: Short Papers, pages 17–20,
Boulder, Colorado, June 2009.

tasks. The triplet lexicon model presented in this work can also be interpreted as an extension of the standard IBM model 1 (Brown et al., 1993) with an additional trigger.

2 Setup

The main focus of this work investigates an extended lexicon model in search and rescoring. The model that we consider here and its integration in the decoder and setup for rescoring are presented in the following sections.

2.1 Extended lexicon model

The triplets of the extended lexicon model $p(e|f, f')$ are composed of two words in the source language triggering one target word. In order to limit the overall number of triplets, we apply a training constraint that reuses the word alignment information obtained in the GIZA++ step. For source words f, we only consider the ones that are aligned to a target word e given the GIZA++ word alignment. The second trigger f' is allowed to move over the whole source sentence, thus capturing long-distance effects that can be observed in the training data:

$$p(e_1^I|f_1^J, \{a_{ij}\}) = \prod_{i=1}^{I} p(e_i|f_1^J, \{a_{ij}\}) = \prod_{i=1}^{I} \frac{1}{Z_i} \sum_{j \in \{a_i\}} \sum_{j'=1}^{J} p(e_i|f_j, f_{j'}) \quad (1)$$

where $\{a_{ij}\}$ denotes the alignment matrix of the sentence pair f_1^J and e_1^I and the first sum goes over all f_j that are aligned to the current e_i (expressed as $j \in \{a_i\}$). The factor $Z_i = J \cdot |\{a_i\}|$ normalizes the double summation accordingly. Eq. 1 is used in the iterative EM training on all sentence pairs of the training data. Empty words are allowed on the triggering part and low probability triplets are trimmed.

2.2 Decoding

Regarding the search, we can apply this model directly when scoring bilingual phrase pairs. Given a trained model for $p(e|f, f')$, we compute the feature score h_t of a phrase pair $(\tilde{e}, \tilde{f})$ as

$$h_t(\tilde{e}, \tilde{f}, \{\tilde{a}_{ij}\}, f_1^J) = -\sum_i \log \sum_{j \in \{\tilde{a}_i\}} \sum_{j'} p(\tilde{e}_i|\tilde{f}_j, f_{j'}) + \sum_i \log Z_i \quad (2)$$

where i moves over all target words in the phrase $\tilde{e}$, the sum over j selects the aligned source words $\tilde{f}_j$ given $\{\tilde{a}_{ij}\}$, the alignment matrix within the phrase pair, and j' incorporates the whole source sentence f_1^J. Analogous to Eq. 1, $Z_i = J \cdot |\{\tilde{a}_i\}|$ denotes the number of overall source words times the number of aligned source words to each $\tilde{e}_i$. In Eq. 2, we take negative log-probabilities and normalize to obtain the final score (representing costs) for the given phrase pair. Note that in search, we can only use this direction, $p(e|f, f')$, since the whole source sentence is available for triggering effects whereas not all target words have been generated so far, as it would be necessary for the reverse direction, $p(f|e, e')$. Due to data sparseness, we smooth the model by using a floor value of 10^{-7} for unseen events during decoding. Furthermore, an implicit backoff to IBM1 exists if the second trigger is the empty word, i.e. for events of the form $p(e|f, \varepsilon)$.

2.3 Rescoring

In rescoring, we constrain the scoring of our hypotheses to a limited set of n-best translations that are extracted from the word graph, a pruned compact representation of the search space. The advantage of n-best list rescoring is the full availability of both source text and target translation, thus allowing for the application of additional (possibly more complex) models that are hard to implement directly in search, such as e.g. syntactic models based on parsers or huge LMs that would not fit in memory during decoding. Since we are limiting ourselves to a small extract of translation hypotheses, rescoring models cannot outperform the same models if applied directly in search. One advantage though is that we can apply the introduced trigger model also in the other direction, i.e. using $p(f|e, e')$, where two target words trigger one source word. Generally, the combination of two directions of a model yields further improvements, so we investigated how this additional direction helps in rescoring (cf. Section 3.1).

In our experiments, we use 10 000-best lists extracted from the word graphs. An initial setting uses the baseline system, whereas a comparative setup incorporates the $(e|f, f')$ direction of the trigger lexicon model in search and adds the reversed direction in rescoring. Additionally, we use n-gram posteriors, a sentence length model and two large language

	train (ch/en)	test08 (NW/WT)	
Sent. pairs	9.1M	480	490
Run. words	259M/300M	14.8K	12.3K
Vocabulary	357K/627K	3.6K	3.2K

Table 1: GALE Chinese-English corpus statistics.

models, a 5-gram count LM trained on 2.5G running words and the Google Web 1T 5-grams. The feature weights of the log-linear mix are tuned on a separate development set using the Downhill Simplex algorithm.

3 Experiments

The experiments are carried out with a GALE system using the official development and test sets of the GALE 2008 evaluation. The corpus statistics are shown in Table 1. The triplet lexicon model was trained on a subset of the overall data. We used 1.4M sentence pairs with 32.3M running words on the English side. The vocabulary sizes were 76.5K for the source and 241.7K for the target language. The final lexicon contains roughly 62 million triplets.

The baseline system incorporates the standard model setup used in phrase-based SMT which combines phrase translation and word lexicon models in both directions, a 5-gram language model, word and phrase penalties, and two models for reordering (a standard distortion model and a discriminative phrase orientation model). For a fair comparison, we also added the related IBM model 1 $p(e|f)$ to the baseline since it can be computed on the sentence-level for this direction, target given source. This step achieves +0.5% BLEU on the development set for newswire but has no effect on test. As will be presented in the next section, the extension to another trigger results in improvements over this baseline, indicating that the extended triplet model is superior to the standard IBM model 1. The feature weights were optimized on separate development sets for both newswire and web text.

We perform the following pipeline of experiments: A first run generates word graphs using the baseline models. From this word graph, we extract 10k-best lists and compare the performance to a reranked version including the additional models. In a second step, we add one of the trigger lexicon models to the search process, regenerate word graphs, extract updated n-best lists and add the remaining models again in a reranking step.

Chinese-English	newswire		web text	
GALE test08	BLEU	TER	BLEU	TER
baseline	32.5	59.4	25.8	64.0
rescore, no triplets	32.8	59.0	26.6	63.5
resc. triplets fe+ef	33.2	58.6	27.1	63.0
triplets in search ef	33.1	58.8	26.0	63.5
rescore, no triplets	33.2	58.6	26.7	63.5
rescore, triplets fe	33.7	58.1	27.2	62.0

Table 2: Results obtained for the two test sets. For the triplet models, "fe" means $p(f|e,e')$ and "ef" denotes $p(e|f,f')$. BLEU/TER scores are shown in percent.

3.1 Results

Table 2 presents results that were obtained on the test sets. All results are based on lowercase evaluations since the system is trained on lowercased data in order to keep computational resources feasible. For the newswire setting, the baseline is 32.5% BLEU and 59.4% TER. Rescoring with additional models not including triplets gives only slight improvements. By adding the path-aligned triplet model in both directions, we observe an improvement of +0.7% BLEU and -0.8% TER. Using the triplet model in source to target direction (e, f, f') during the search process, we arrive at a similar BLEU improvement of +0.6% without any reranking models. We add the other direction of the triplets (f, e, e') (the one that can not be used directly in search) and obtain 33.7% BLEU on the newswire set. The overall cumulative improvements of triplets in search and reranking are +0.9% BLEU and -0.9% TER when compared to the rescored baseline not incorporating triplet models and +1.2%/-1.3% on the decoder baseline, respectively.

For the web text setting, the baseline is considerably lower at 25.8% BLEU and 64.0% TER (cf. right part of Table 2). We observe an improvement for the baseline reranking models, a large part of which is due to the Google Web LM. Adding triplets to search does not help significantly (+0.2%/-0.5% BLEU/TER). This might be due to training the triplet lexicon mainly on newswire data. Reranking without triplets performs similar to the baseline

experiment. Mixing in the (f, e, e') direction helps again: The final score comes out at 27.2% BLEU and 62.0% TER, the latter being significantly better than the reranked baseline (-1.5% in TER).

3.2 Discussion

The results indicate that it is worth moving models from rescoring to the search process. This is not surprising (and probably well known in the community). Interestingly, the triplet model can improve translation quality in addition to its related IBM model 1 which was already part of the baseline. It seems that the extension by a second trigger helps to capture some language specific properties for Chinese-English which go beyond local lexical (word-to-word) dependencies. In Table 3, we show an example of improved translation quality where a triggering effect can be observed. Due to the topic of the sentence, the phrase *local employment* was chosen over *own jobs*. One of the top triplets in this context is $p(\text{employment} \mid \text{就业}, \text{人才})$, where 就业 is "employment" due to the path-aligned constraint and 人才 means "talent". Note that the distance between these two triggers is five tokens.

source	德国 为了 保护 本国 人 就业 , 对 引进 国外 人才 设 了 较 高 的 门槛 .
baseline	germany, in order to protect their own jobs, the introduction of foreign talent, a relatively high threshold.
triplets	in order to protect local employment, germany has a relatively high threshold for the introduction of foreign talent.
reference	in order to protect native employment, germany has set a relatively high threshold for bringing in foreign talents.

Table 3: Translation example on the newswire test set.

4 Conclusion

We presented the integration of an extended lexicon model into the search process and compared it to a variant which was used in reranking n-best lists. In order to keep the overall number of triplets feasible, and thus memory footprints and training times low, we chose a path-constrained triplet model that restricts the first source trigger to the aligned target word, whereas the second trigger can move along the whole source sentence. The motivation was to allow for a more fine-grained lexical choice of target words by looking at sentence-level context. The overall improvements that can be accounted to the triplets are up to +0.9% BLEU and -1.5% TER.

In the future, we plan to investigate more triplet model variants and work on additional language pairs such as French-English or German-English. The reverse direction, $p(f|e, e')$, is hard to implement outside of a reranking framework where the full target hypotheses are already fully generated. It might be worth looking at cross-lingual trigger models such as $p(f|e, f')$ or constrained variants like $p(f|e, e')$ with $e' < e$, i.e. the second trigger coming from the left context within a sentence which has already been generated.

Acknowledgments

This material is partly based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR0011-06-C-0023, and was partly realized as part of the Quaero Programme, funded by OSEO, French State agency for innovation.

The authors would like to thank Juri Ganitkevitch for training the triplet model.

References

P. F. Brown, 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):263–311, June.

M. Carpuat and D. Wu. 2007. Improving statistical machine translation using word sense disambiguation. In *Proc. EMNLP-CoNLL*, Prague, Czech Republic, June.

Y. S. Chan, H. T. Ng, and D. Chiang. 2007. Word sense disambiguation improves statistical machine translation. In *Proc. ACL*, pages 33–40, Prague, Czech Republic, June.

I. García-Varea, F. J. Och, H. Ney, and F. Casacuberta. 2001. Refined lexicon models for statistical machine translation using a maximum entropy approach. In *Proc. ACL Data-Driven Machine Translation Workshop*, pages 204–211, Toulouse, France, July.

S. Hasan, J. Ganitkevitch, H. Ney, and J. Andrés-Ferrer. 2008. Triplet lexicon models for statistical machine translation. In *Proc. EMNLP*, pages 372–381, Honolulu, Hawaii, October.

D. Vickrey, L. Biewald, M. Teyssier, and D. Koller. 2005. Word-sense disambiguation for machine translation. In *Proc. HLT-EMNLP*, pages 771–778, Morristown, NJ, USA.

A Simplex Armijo Downhill Algorithm for Optimizing Statistical Machine Translation Decoding Parameters

Bing Zhao
IBM T.J. Watson Research
zhaob@us.ibm.com

Shengyuan Chen
IBM T.J. Watson Research
sychen@us.ibm.com

Abstract

We propose a variation of simplex-downhill algorithm specifically customized for optimizing parameters in statistical machine translation (SMT) decoder for better end-user automatic evaluation metric scores for translations, such as versions of BLEU, TER and mixtures of them. Traditional simplex-downhill has the advantage of derivative-free computations of objective functions, yet still gives satisfactory searching directions in most scenarios. This is suitable for optimizing translation metrics as they are not differentiable in nature. On the other hand, Armijo algorithm usually performs line search efficiently given a searching direction. It is a deep hidden fact that an efficient line search method will change the iterations of simplex, and hence the searching trajectories. We propose to embed the Armijo inexact line search within the simplex-downhill algorithm. We show, in our experiments, the proposed algorithm improves over the widely-applied Minimum Error Rate training algorithm for optimizing machine translation parameters.

1 Introduction

A simple log-linear form is used in SMT systems to combine feature functions designed for identifying good translations, with proper weights. However, we often observe that tuning the weight associated with each feature function is indeed not easy. Starting from a N-Best list generated from a translation decoder, an optimizer, such as Minimum Error Rate (MER) (Och, 2003) training, proposes directions to search for a better *weight-vector* λ to combine feature functions. With a given λ, the N-Best list is re-ranked, and newly selected top-1 hypothesis will be used to compute the final MT evaluation metric score. Due to limited variations in the N-Best list, the nature of ranking, and more importantly, the *non-differentiable* objective functions used for MT (such as BLEU (Papineni et al., 2002)), one often found only local optimal solutions to λ, with no clue to walk out of the riddles.

Automatic evaluation metrics of translations known so far are designed to simulate human judgments of translation qualities especially in the aspects of *fluency* and *adequacy*; they are not differentiable in nature. Simplex-downhill algorithm (Nelder and Mead, 1965) does not require the objective function to be differentiable, and this is well-suited for optimizing such automatic metrics. MER searches *each* dimension independently in a *greedy* fashion, while simplex algorithms consider the movement of *all* the dimensions at the same time via three basic operations: *reflection*, *expansion* and *contraction*, to shrink the simplex iteratively to some local optimal. Practically, as also shown in our experiments, we observe simplex-downhill usually gives better solutions over MER with *random restarts* for both, and reaches the solutions much faster in most of the cases. However, simplex-downhill algorithm is an *unconstrained* algorithm, which does not leverage any domain knowledge in machine translation. Indeed, the objective function used in SMT is shown to be a piece-wise linear problem in (Papineni et al., 1998), and this motivated us to embed an inexact line search with Armijo rules (Armijo, 1966) within a simplex to guide the directions for iterative expansion, reflection and contraction operations. Our proposed modification to the simplex algorithm is an embedded backtracking line search, and the algorithm's convergence (McKinnon, 1999) still holds, though it is configured specially here for optimizing automatic machine translation evaluation metrics.

The remainder of the paper is structured as follow: we briefly introduce the optimization problem in section 2; in section 3, our proposed simplex Armijo downhill algorithm is explained in details; experiments comparing relevant algorithms are in section 4; the conclusions and discussions are given in section 5.

2 Notations

Let $\{(e_{i,k}, \bar{c}_{i,k}, S_{i,k}), k \in [1, K]\}$ be the K-Best list for a given input source sentence f_i in a development dataset containing N sentences. $e_{i,k}$ is a English hypothesis at the rank of k; $\bar{c}_{i,k}$ is a cost vector — a vector of feature function values, with M dimensions: $\bar{c}_{i,k} = (c_{i,k,1}, c_{i,k,2} \ldots c_{i,k,M})$; $S_{i,k}$ is a *sentence-level* translation metric *general counter* (e.g. ngram hits for BLEU, or specific types of errors counted in TER, etc.) for the hypothesis. Let $\bar{\lambda}$ be the weight-vector, so that the cost of $e_{i,k}$ is an inner product: $C(e_{i,k}) = \bar{\lambda} \cdot \bar{c}_{i,k}$. The optimization process is then defined as below:

$$k^*(\text{wrt } i) = \arg\min_k \bar{\lambda} \cdot \bar{c}_{i,k} \quad (1)$$

$$\bar{\lambda}^* = \arg\min_{\bar{\lambda}} \text{Eval}(\sum_{i=1}^{N} S_{i,k^*}), \quad (2)$$

Proceedings of NAACL HLT 2009: Short Papers, pages 21–24,
Boulder, Colorado, June 2009.

where Eval is an evaluation *Error* metric for MT, presuming the *smaller* the better internal to an optimizer; in our case, we decompose BLEU, TER (Snover et al., 2006) and (TER-BLEU)/2.0 into corresponding specific counters for each sentence, cache the intermediate counts in $S_{i,k}$, and compute final corpus-level scores using the sum of all counters; Eqn. 1 is simply a ranking process, with regard to the source sentence i, to select the *top-1* hypothesis, indexed by k^* with the lowest cost $C(e_{i,k^*})$ given current $\bar{\lambda}$; Eqn. 2 is a scoring process of computing the final corpus-level MT metrics via the intermediate counters collected from each top1 hypothesis selected in Eqn. 1. Iteratively, the optimizer picks up an initial guess of $\bar{\lambda}$ using current K-Best list, and reaches a solution $\bar{\lambda}^*$, and then updates the event space with *new* K-Best list generated using a decoder with $\bar{\lambda}^*$; it iterates until there is little change to final scores (a local optimal $\bar{\lambda}^*$ is reached).

3 Simplex Armijo Downhill

We integrate the Armijo line search into the simplex-downhill algorithm in Algorithm 1. We take the *reflection*, *expansion* and *contractions* steps[1] from the simplex-downhill algorithm to find a λ' to form a direction $\lambda' - \lambda_{M+1}$ as the input to the Armijo algorithm, which in turn updates λ' to λ^+ as the input for the next iteration of simplex-downhill algorithm. The combined algorithm iterates until the simplex shrink sufficiently within a predefined threshold. Via Armijo algorithm, we avoid the expensive *shrink* step, and slightly speed up the searching process of simplex-downhill algorithm. Also, the simplex-downhill algorithm usually provides a descend direction to start the Armijo algorithm efficiently. Both algorithms are well known to converge. Moreover, the new algorithm changes the searching path of the traditional simplex-downhill algorithm, and usually leads to better local minimal solutions.

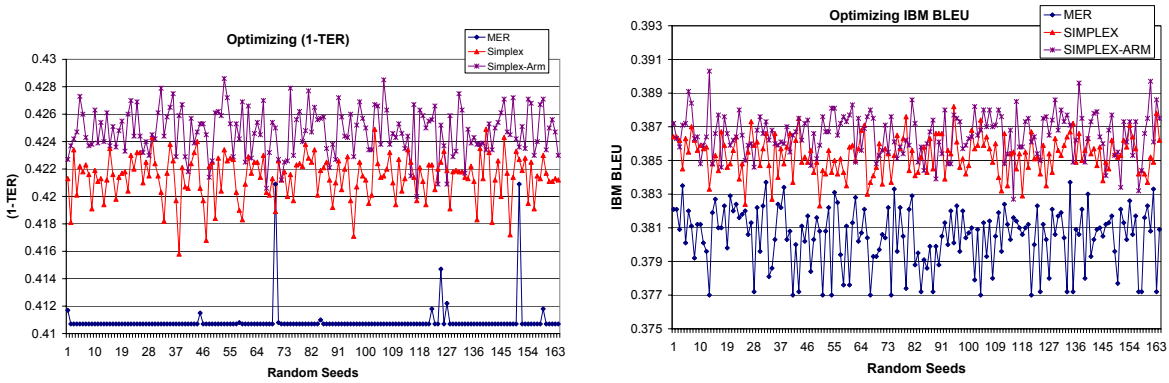
To be more specific, Algorithm 1 clearly conducts an iterative search in the *while* loop from *line* 3 to *line* 28 until the stopping criteria on line 3 is satisfied. Within the loop, the algorithm can be logically divided into two major parts: from line 4 to line 24, it does the simplex-downhill algorithm; the rest does the Armijo search. The simplex-downhill algorithm looks for a lower point by trying the reflection (line 6), expansion (line 10) and contraction (line 17) points in the order showed in the algorithm, which turned out to be very efficient. In rare cases, especially for many dimensions (for instance, 10 to 30 dimensions, as in typical statistical machine translation decoders) none of these three points are not lower enough (line 21), we adapt other means to select lower points. We avoid the traditional expensive shrink procedure, which is not favorable for our machine translation problem neither. Instead we try points on different search lines. Specifically, we test *two* additional points on the line through the highest point and the lowest point, and on the line through the reflection point and the lowest point. It worth pointing out that there are many variants of simplex-downhill algorithm [2], and the implementation described above showed that the algorithm can successfully select a lower λ' in many of our translation test cases to enable the simplex move to a better region of local optimals in the high-dimension space. Our proposed embedded Armijo algorithm, in the second part of the loop (line 25), continues to *refine* the search processes. By backtracking on the segment from λ' to λ_{M+1}, the Armijo algorithm does bring even lower points in our many test cases. With the new lower λ' found by the Armijo algorithm, the simplex-downhill algorithm starts over again. The parameters in line 1 we used are com-

Algorithm 1 Simplex Armijo Downhill Algorithm

1: $\alpha \Leftarrow 1, \gamma \Leftarrow 2, \rho \Leftarrow 0.5, \beta = \eta \Leftarrow 0.9, \epsilon \Leftarrow 1.0 \times 10^{-6}$
2: initilize $(\lambda_1, \cdots, \lambda_{M+1})$
3: **while** $\sum_{i,j=1}^{M+1} \|\lambda_i - \lambda_j\|_2 \leq \epsilon$ **do**
4: sort λ_i ascend
5: $\lambda_o \Leftarrow \frac{1}{N} \sum_{i=1}^{M} \lambda_i$,
6: $\lambda_r \Leftarrow \lambda_o + \alpha(\lambda_o - \lambda_{M+1})$
7: **if** $S(\lambda_1) \leq S(\lambda_r) \leq S(\lambda_M)$ **then**
8: $\lambda' \Leftarrow \lambda_r$
9: **else if** $S(\lambda_r) < S(\lambda_1)$ **then**
10: $\lambda_e \Leftarrow \lambda_o + \gamma(\lambda_o - \lambda_{M+1})$
11: **if** $S(\lambda_e) < S(\lambda_r)$ **then**
12: $\lambda' \Leftarrow \lambda_e$
13: **else**
14: $\lambda' \Leftarrow \lambda_r$
15: **end if**
16: **else if** $S(\lambda_r) > S(\lambda_M)$ **then**
17: $\lambda_c \Leftarrow \lambda_{M+1} + \rho(\lambda_o - \lambda_{M+1})$
18: **if** $S(\lambda_c) < S(\lambda_r)$ **then**
19: $\lambda' \Leftarrow \lambda_c$
20: **else**
21: try points on two additional lines for λ'
22: **end if**
23: **end if**
24: $d \Leftarrow \lambda' - \lambda_{M+1}$
25: $\beta^* \Leftarrow max_{k=0,1,\cdots,40}\{\beta^k | S(\lambda_{M+1} + \beta^k d) - S(\lambda_{M+1}) \leq -\eta \|d\|_2 \beta^k\}$
26: $\lambda^+ = \lambda_{M+1} + \beta^* * d$
27: replace λ_{M+1} with λ^+
28: **end while**

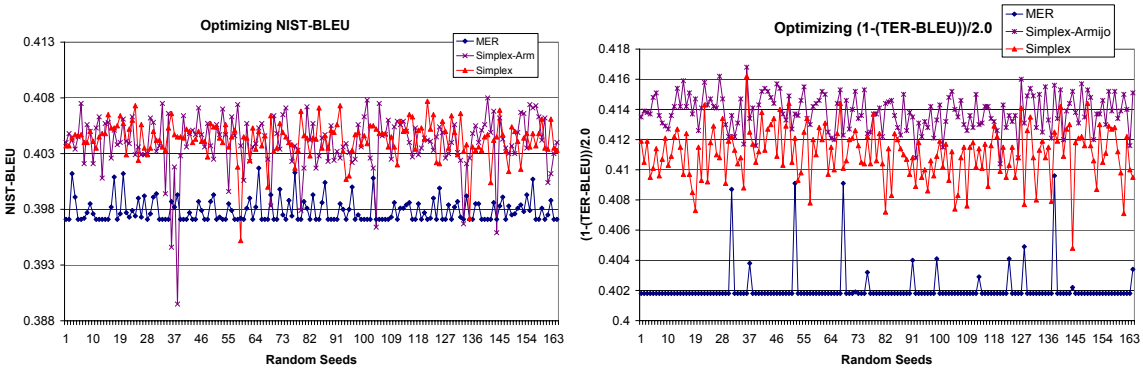
[1] These three basic operations are generally based on heuristics in the traditional simplex-downhill algorithm.

[2] One of such effective tricks for the baseline simplex algorithms can be found here: http://paula.univ.gda.pl/∼dokgrk/simplex.html (link tested to be valid as of 04/03/2009)



(a) Optimizing Toward Metric (1-TER)

(b) Optimizing Toward Metric IBM BLEU



(c) Optimizing Toward Metric NIST BLEU

(d) Optimizing Toward Metric (1-(TER-NISTBLEU))/2

Figure 1: On devset, comparing MER, Simplex Downhill, and Simplex Armijo Downhill Algorithms on different Translation Metrics including TER, IBM BLEU, NIST BLEU, and the combination of TER & NISTBLEU. Empirically, we found optimizing toward (TER-NISTBLEU)/2 gave more reliable solutions on unseen test data. All optimizations are with internal random restarts, and were run from the same 164 random seeds with *multiple iterations* until convergence. Simplex Armijo downhill algorithm is often better than Simplex-downhill algorithm, and is also much better than MER algorithm.

mon ones from literatures and can be tuned further. We find that the combination not only accelerates the searching process to reach similar solutions to the baseline simplex algorithm, but also changes the searching trajectory significantly, leading to even better solutions for machine translation test cases as shown in our experiments.

4 Experiments

Our experiments were carried out on Chinese-English using our syntax-based decoder (Zhao and Al-Onaizan, 2008), a chart-based decoder with tree-to-string [3] grammar, in GALE P3/P3.5 evaluations. There were 10 feature functions computed for each hypothesis, and N-best list size is up to 2,000 per sentence.

Given a weight-vector $\bar{\lambda}_0$, our decoder outputs N-Best *unique* hypotheses for each input source sentence; the event space is then built, and the optimizer is called with a number of random restarts. We used 164 seeds[4] with a small perturbation of three random dimensions in $\bar{\lambda}_0$. The best $\bar{\lambda}_1$ is selected under a given optimizing metric, and is fed back to the decoder to re-generate a *new* N-Best list. Event space is enriched by merging the newly generated N-Best list, and the optimization runs again. This process is iteratively carried out until there are no more improvements observed on a development data set.

We select *three* different metrics: NIST BLEU, IBM BLEU, TER, and a combination of (TER-NISTBLEU)/2 as our optimization goal. On the devset with four references using MT06-NIST text part data, we carried out the optimizations as shown in Figure 1. Over these 164 random restarts in each of the optimizers over the four configurations shown in Figure 1, we found most of the time simplex algorithms perform better than MER in these configurations. Simplex algorithm considers to move all the dimensions at the same time, instead of fixing other

[3]Source shallow constituency tree to target-string rules with variables, forming a probabilistic synchronous context free grammar.

[4]There are 41 servers used in our experiments, four CPUs each.

Table 1: Comparing different optimization algorithms on the held-out speech data, measured on document-average TER, IBMBLEU and (TER-IBMBLEU)/2.0, which were used in GALE P3/3.5 Chinese-English evaluations in Rosetta consortium.

Setup	Broadcast News & Conversation Data		
	BLEUr4n4	TER	(TER-BLEUr4n4)/2
MER	37.36	51.12	6.88
Simplex-Downhill	37.71	50.10	6.19
Simplex Armijo Downhill	38.15	49.92	5.89

dimensions and carrying out a greedy search for one dimension as in MER. With Armijo line search embedded in the simplex-downhill algorithm, the algorithm has a better chance to walk out of the local optimal, via changing the shrinking trajectory of the simplex using a line search to identify the best steps to move. Shown in Figure 1, the solutions from simplex Armijo downhill outperformed the other two under four different optimization metrics for most of the time. Empirically, we found optimizing toward (TER-NISTBLEU)/2 gives marginally better results on final TER and IBM BLEU.

On our devset, we also observed that whenever optimizing toward TER (or mixture of TER & BLEU), MER does not seem to move much, as shown in Figure 1-(a) and Figure 1-(d). However, on BLEU (NIST or IBM version), MER does move reasonably with random restarts. Comparing TER with BLEU, we think the "*shift*" counter in TER is a *confusing* factor to the optimizer, and cannot be computed accurately in the current TER implementations. Also, our random perturbations to the seeds used in restarts might be relatively weaker for MER comparing to our simplex algorithms, though they use exactly the same random seeds. Another fact we found is optimizing toward corpus-level (TER-NISTBLEU)/2 seems to give better performances on most of our unseen datasets, and we choose this as optimization goal to illustrate the algorithms' performances on our unseen testset.

Our test set is the held-out speech part data[5]. We optimize toward corpus-level (TER-NISTBLEU)/2 using devset, and apply the weight-vector on testset to evaluate TER, IBMBLEUr4n4, and a simple combination of (TER-IBMBLEU)/2.0 to compare different algorithms' strengths[6]. Shown in Table 1, simplex Armijo downhill performs the best (though not statistically significant), and the improvements are *consistent* in multiple runs in our observations. Also, given limited resources, such as number of machines and fixed time schedule, both simplex algorithms can run with more random restarts than MER, and can potentially reach better solutions.

[5]Transcriptions of broadcast news and broadcast conversion in MT06; there are 565 sentences, or 11,691 words after segmentation.

[6]We choose document-average metrics to show here simply because they were chosen/required in our GALE P3/P3.5 evaluations for both Arabic-English and Chinese-English individual systems and syscombs.

5 Conclusions and Discussions

We proposed a simplex Armijo downhill algorithm for improved optimization solutions over the standard simplex-downhill and the widely-applied MER. The Armijo algorithm changes the trajectories for the simplex to shrink to a local optimal, and empowers the algorithm a better chance to walk out of the riddled error surface computed by automatic MT evaluation metrics. We showed, empirically, such utilities under several evaluation metrics including BLEU, TER, and a mixture of them. In the future, we plan to integrate domain specific heuristics via approximated derivatives of evaluation metrics or mixture of them to guide the optimizers move toward better solutions for simplex-downhill algorithms.

References

L. Armijo. 1966. Minimization of functions having lipschitz continuous first partial derivatives. *Pacific Journal of mathematics*, 6:1–3.

K.I.M. McKinnon. 1999. Convergence of the nelder-mead simplex method to a non-stationary point. *SIAM J Optimization*, 9:148–158.

J.A. Nelder and R. Mead. 1965. A simplex method for function minimization. *The Computer Journal*, 7:308–313.

Franz J. Och. 2003. Minimum error rate training for statistical machine translation. In *Proc. of the 41st Annual Meeting of the Association for Computational Linguistics*, Japan, Sapporo, July.

Kishore Papineni, Salim Roukos, and Todd Ward. 1998. Maximum likelihood and discriminative training of direct translation models. In *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech & Signal Processing*, volume 1, pages 189–192, Seattle, May.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of the 40th Annual Conf. of the Association for Computational Linguistics (ACL 02)*, pages 311–318, Philadelphia, PA, July.

Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *AMTA*.

Bing Zhao and Yaser Al-Onaizan. 2008. Generalizing local and non-local word-reordering patterns for syntax-based machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Translation Corpus Source and Size in Bilingual Retrieval

Paul McNamee and **James Mayfield**
Human Language Technology Center of Excellence
Johns Hopkins University
Baltimore, MD 21218, USA
{paul.mcnamee,james.mayfield}@jhuapl.edu

Charles Nicholas
Dept. of Computer Science and Electrical Engineering
UMBC
Baltimore, MD 21250, USA
nicholas@umbc.edu

Abstract

This paper explores corpus-based bilingual retrieval where the translation corpora used vary by source and size. We find that the quality of translation alignments and the domain of the bitext are important. In some settings these factors are more critical than corpus size. We also show that judicious choice of tokenization can reduce the amount of bitext required to obtain good bilingual retrieval performance.

1 Introduction

Large parallel corpora are an increasingly available commodity. Such texts are the fuel of statistical machine translation systems and are used in applications such as cross-language information retrieval (CLIR). Several beliefs are commonly held regarding the relationship between parallel text *quality* and *size* for CLIR. It is thought that larger texts should be better, because the problems of data sparseness and untranslatable terms are reduced. Similarly, parallel text from a domain more closely related to a document collection should lead to better bilingual retrieval performance, again because better lexical translations are available.

We compared four sources of parallel text using CLEF document collections in eight languages (Braschler and Peters, 2004). English topic sets from 2000 to 2007 were used. Corpus-based translation of query terms was performed and documents were ranked using a statistical language model approach to retrieval (Ponte and Croft, 1998). Experiments were conducted using unlemmatized words and character 5-grams. No use was made of pre-translation query expansion or automated relevance feedback.

2 Translation Corpora

Information about the four parallel texts used in our experiments is provided in Table 1. We restricted our focus to Dutch (NL), English (EN), Finnish (FI), French (FR), German (DE), Italian (IT), Portuguese (PT), Spanish (ES), and Swedish (SV). These languages are covered by each parallel corpus.

2.1 Bible

The *bible* corpus is based on the 66 books in the Old and New Testaments. Alignments at the verse level were used; there are 31103 verses in the English text.

2.2 JRC-Acquis v3

This parallel text is based on EU laws comprising the *Acquis Communautaire* and translations are available in 22 languages. The English portion of the *acquis* data includes 1.2 million aligned passages containing over 32 million words, which is approximately 40 times larger than the Biblical text. Alignments were provided with the corpus and were produced by the *Vanilla* algorithm.[1] The alignments are at roughly the sentence level, but only 85% correspond to a single sentence in both languages.

2.3 Europarl v3

The Europarl corpus was assembled to support experiments in statistical machine translation (Koehn, 2005). The documents consist of transcribed dialogue from the official proceedings of the European Parliament. We used the precomputed alignments that are provided with the corpus, and which are based on the algorithm by Gale and Church (1991). The alignments are believed to be of high quality.

[1] Available from http://nl.ijs.si/telri/vanilla/

Proceedings of NAACL HLT 2009: Short Papers, pages 25–28,
Boulder, Colorado, June 2009.

Name	Words	Wrds/doc	Alignments	Genre	Source
bible	785k	25.3	Near Perfect	Religious	http://unbound.biola.edu/
acquis	32M	26.3	Good	EU law (1958 to 2006)	http://wt.jrc.it/lt/acquis/
europarl	33M	25.5	Very Good	Parliamentary oration (1996 to 2006)	http://www.statmt.org/europarl/
ojeu	84M	34.5	Fair	Governmental affairs (1998 to 2004)	Derived from documents at http://europea.eu.int/

Table 1: Parallel texts used in experiments.

2.4 Official Journal of the EU

The Official Journal of the European Union covers a wide range of topics such as agriculture, trade, and foreign relations. We constructed this parallel corpus by downloading documents dating from January 1998 through April 2004 and converting the texts from Adobe's Portable Document Format (PDF) to ISO-8859-1 encoded text using *pdftotext*. The documents were segmented into pages and into paragraphs consisting of a small number of sentences (typically 1 to 3); however this process was complicated by the fact that many documents have outline or tabular formatting. Alignments were produced using Church's *char_align* software (1993).

Due to complexities of decoding the PDF, some of the accented characters were not extracted properly, but this is a problem mostly for the earlier material in the collection. In total about 85 million words of text per language was obtained, which is over twice the size of either the *acquis* or *europarl* collections.

3 Translation

Using the pairwise-aligned corpora described above, parallel indexes for each corpus were created using words and 5-grams. Query translation was accomplished as follows. For each query term s, source language documents from the aligned collection that contain s are identified. If no document contains this term, then it is left untranslated. Each target language term t appearing in the corresponding documents is scored:

$$Score(t) = (F_l(t) - F_c(t)) \times IDF(t)^{1.25} \quad (1)$$

where F_l and F_c are relative document frequencies based on local subset of documents and the whole corpus. $IDF(t)$ is the inverse document frequency, or $log_2(\frac{N}{df(t)})$. The candidate translation with the highest score replaced the original query term and the transformed query vector is used for retrieval against the target language collection.

This is a straightforward approach to query translation. More sophisticated methods have been proposed, including bidirectional translation (Wang and Oard, 2006) and use of more than one translation candidate per query term (Pirkola et al., 2003).

Subword translation, the direct translation of character n-grams, offers several advantages over translating words (McNamee and Mayfield, 2005). N-grams provide morphological normalization, translations of multiword expressions are suggested by translation of word-spanning n-grams, and out-of-vocabulary (OOV) words can be be partly translated with n-gram fragments. Additionally, there are few OOV n-grams, at least for $n = 4$ and $n = 5$.

4 Experimental Results

We describe two experiments. The first examines the efficacy of the different translation resources and the second measures the relationship between corpus size and retrieval effectiveness. English was the sole source language.

4.1 Translation Resources

First the relationship between translation source and bilingual retrieval effectiveness is studied. Table 2 reports mean average precision when word-based tokenization and translation was performed for each of the target collections. For comparison the corresponding performance using topics in the target language (*mono*) is also given. As expected, the smallest bitext, *bible*, performs the worst. Averaged across the eight languages only 39% relative effectiveness is seen compared to monolingual performance. Reports advocating the use of religious texts for general purpose CLIR may have been overly optimistic (Chew et al., 2006). Both *acquis* and *europarl* are roughly 40 times larger in size than *bible*

Target	*mono*	*bible*	*acquis*	*europarl*	*ojeu*
DE	0.3303	0.1338	0.1802	0.2427	0.1937
ES	0.4396	0.1454	0.2583	0.3509	0.2786
FI	0.3406	0.1288	0.1286	0.2135	0.1636
FR	0.3638	0.1651	0.2508	0.2942	0.2600
IT	0.3749	0.1080	0.2365	0.2913	0.2405
NL	0.3813	0.1502	0.2474	0.2974	0.2484
PT	0.3162	0.1432	0.2009	0.2365	0.2157
SV	0.3387	0.1509	0.2111	0.2447	0.1861
Average	0.3607	0.1407	0.2142	0.2714	0.2233
		39.0%	59.4%	75.3%	61.9%

Table 2: Mean average precision for word-based translation of English topics using different corpora.

Target	*mono*	*bible*	*acquis*	*europarl*	*ojeu*
DE	0.4201	0.1921	0.2952	0.3519	0.3169
ES	0.4609	0.2295	0.3661	0.4294	0.3837
FI	0.5078	0.1886	0.3552	0.3744	0.3743
FR	0.3930	0.2203	0.3013	0.3523	0.3334
IT	0.3997	0.2110	0.2920	0.3395	0.3160
NL	0.4243	0.2132	0.3060	0.3603	0.3276
PT	0.3524	0.1892	0.2544	0.2931	0.2769
SV	0.4271	0.1653	0.3016	0.3203	0.2998
Average	0.4232	0.2012	0.3090	0.3527	0.3286
		47.5%	73.0%	83.3%	77.6%

Table 3: Mean average precision using 5-gram translations of English topics using different corpora.

and both do significantly better; however *europarl* is clearly superior and achieves 75% of monolingual effectiveness. Though nearly twice the size, *ojeu* fails to outperform *europarl* and just barely beats *acquis*. Likely reasons for this include difficulties properly converting the *ojeu* data to text, problematic alignments, and the substantially greater length of the aligned passages.

The same observations can be seen from Table 3 where 5-grams were used for tokenization and translation instead of words. The level of performance with 5-grams is higher and these improvements are statistically significant with $p < 0.01$ (*t*-test).[2] Averaged across the eight languages gains from 30% to 47% were seen using 5-grams, depending on the resource. As a translation resource *europarl* still outperforms the other sources in each of the eight languages and the relative ordering of {*europarl, ojeu, acquis, bible*} is the same in both cases.

[2]Except in four cases: *mono*: In ES & IT $p < 0.05$; *bible*: 5-grams were not significantly different than words in FI & SV

4.2 Size of Parallel Text

To investigate how corpus size effects bilingual retrieval we subsampled *europarl* and used these smaller subcorpora for translation. The entire corpus is 33 million words in size, and samples of 1%, 2%, 5%, 10%, 20%, 40%, 60%, and 80% were made based on counting documents, which for *europarl* is equivalent to counting sentences. Samples were taken by processing the data in chronological order.

In Figure 1 (a-d) the effect of using larger parallel corpora is plotted for four languages. Mean average precision is on the vertical axes, and for visual effect the chart for each language pair uses the same scale. The general shape of the curves is to rise quickly as increasing subsets from 1% to 10% are used and to flatten as size increases further. Curves for the other four languages (not shown) are quite similar. The deceleration of improvement with increasing corpus size can be explained by Heap's Law. Similar results have been obtained in the few studies that have sought to quantify bilingual retrieval performance as a function of translation resource size (Xu and Weischedel, 2000; Demner-Fushman and Oard, 2003). In the higher complexity languages such as German and Finnish, n-grams appear to be gaining a slight improvement even when the entire corpus is used; vocabulary size is greater in those languages.

The data for the 0% condition were based on cognate matches for words and 'cognate n-grams' that require no translation. The figure reveals that even very small amounts of parallel text quickly improve performance. The 2% condition is roughly the size of *bible*, but is higher performing, likely due to a better domain match.[3] Using a subsample of only 5% of available data from the highest performing translation resource, *europarl*, 5-grams outperformed plain words using any amount of bitext.

5 Conclusion

We examined issues in corpus-based bilingual retrieval, including the importance of parallel corpus selection and size, and the relative effectiveness of alternative tokenization methods. Size is not the only important factor in corpus-based bilingual re-

[3]For example, the Biblical text does not contain the words *nuclear* or *energy* and thus is greatly disadvantaged for a topic about nuclear power.

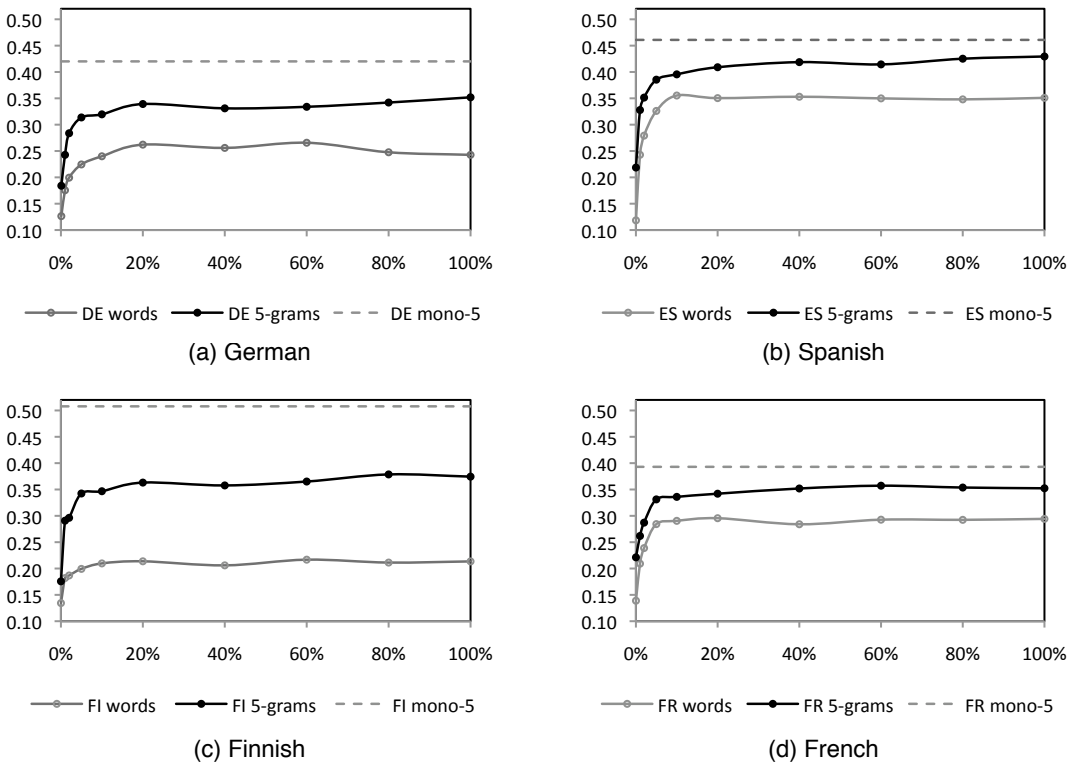


Figure 1: Performance improvement with corpus growth.

trieval, the quality of alignments, compatibility in genre, and choice of tokenization are also important.

We found that character 5-gram tokenization outperforms words when used both for translation and document indexing. Large relative improvements (over 30%) were observed with 5-grams, and when only limited parallel data is available for translation, n-grams are markedly more effective than words.

Future work could address some limitations of the present study by using bidirectional translation models, considering other language families and source languages other than English, and applying query expansion techniques.

References

Martin Braschler and Carol Peters. 2004. Cross-language evaluation forum: Objectives, results, achievements. *Inf. Retr.*, 7(1-2):7–31.

P. A. Chew, S. J. Verzi, T. L. Bauer, and J. T. McClain. 2006. Evaluation of the Bible as a resource for cross-language information retrieval. In *Workshop on Multilingual Language Resources and Interoperability*, pages 68–74.

Kenneth Ward Church. 1993. Char_align: A program for aligning parallel texts at the character level. In *Proceedings ACL*, pages 1–8.

Dina Demner-Fushman and Douglas W. Oard. 2003. The effect of bilingual term list size on dictionary-based cross-language information retrieval. In *HICSS*, pages 108–117.

William A. Gale and Kenneth W. Church. 1991. A program for aligning sentences in bilingual corpora. In *Proceedings ACL*, pages 177–184.

Philipp Koehn. 2005. Europarl: A parallel corpus for statistical machine translation. In *MT Summit*.

Paul McNamee and James Mayfield. 2005. Translating pieces of words. In *ACM SIGIR*, pages 643–644.

Ari Pirkola, Deniz Puolamäki, and Kalervo Järvelin. 2003. Applying query structuring in cross-language retrieval. *Inf. Process. Manage*, 39(3):391–402.

Jay M. Ponte and W. Bruce Croft. 1998. A language modeling approach to information retrieval. In *ACM SIGIR*, pages 275–281.

Jianqiang Wang and Douglas W. Oard. 2006. Combining bidirectional translation and synonymy for cross-language information retrieval. In *ACM SIGIR*, pages 202–209.

Jinxi Xu and Ralph Weischedel. 2000. Cross-lingual information retrieval using hidden Markov models. In *EMNLP*, pages 85–103.

Large-scale Computation of Distributional Similarities for Queries

Enrique Alfonseca
Google Research
Zurich, Switzerland
ealfonseca@google.com

Keith Hall
Google Research
Zurich, Switzerland
kbhall@google.com

Silvana Hartmann
University of Stuttgart
Stuttgart, Germany
silvana.hartmann@ims.uni-stuttgart.de

Abstract

We present a large-scale, data-driven approach to computing distributional similarity scores for queries. We contrast this to recent web-based techniques which either require the off-line computation of complete phrase vectors, or an expensive on-line interaction with a search engine interface. Independent of the computational advantages of our approach, we show empirically that our technique is more effective at ranking query alternatives that the computationally more expensive technique of using the results from a web search engine.

1 Introduction

Measuring the semantic similarity between queries or, more generally, between pairs of very short texts, is increasingly receiving attention due to its many applications. An accurate metric of query similarities is useful for *query expansion*, to improve recall in Information Retrieval systems; for *query suggestion*, to propose to the user related queries that might help reach the desired information more quickly; and for *sponsored search*, where advertisers bid for keywords that may be different but semantically equivalent to user queries.

In this paper, we study the problem of measuring similarity between queries using corpus-based unsupervised methods. Given a query q, we would like to rank all other queries according to their similarity to q. The proposed approach compares favorably to a state-of-the-art unsupervised system.

2 Related work

Distributional similarity methods model the similarity or relatedness of words using a metric defined over the set of contexts in which the words appear (Firth, 1957). One of the most common representations for contexts is the vector space model (Salton et al., 1975). This is the basic idea of approaches such as (Grefenstette, 1992; Bordag, 2008; Lin, 1998; Riloff and Shepherd, 1997), with some variations; e.g., whether syntactic information is used explicitly, or which weight function is applied. Most of the existing work has focused on similarity between single words or syntactically-correct multiword expressions. In this work, we adapt these techniques to calculate similarity metrics between pairs of complete queries, which may or may not be syntactically correct.

Other approaches for query similarity use statistical translation models (Riezler et al., 2008), analysing search engine logs (Jones et al., 2006), looking for different anchor texts pointing to the same pages (Kraft and Zien, 2004), or replacing query words with other words that have the highest pointwise mutual information (Terra and Clarke, 2004).

Sahami and Helman (Sahami and Heilman, 2006) define a web kernel function for semantic similarity based on the snippets of the search results returned by the queries. The algorithm used is the following: (a) Issue a query x to a search engine and collect the set of n snippets returned by the search engine; (b) Compute the tf·idf vector v_i for each document snippet d_i; (c) Truncate each vector to include its m

Proceedings of NAACL HLT 2009: Short Papers, pages 29–32,
Boulder, Colorado, June 2009.

highest weighted terms; (d) Construct the centroid of the L_2-normalized vectors v_i; (e) Calculate the similarity of two queries as the dot product of their L_2-normalized vectors, i.e. as the cosine of both vectors.

This work was followed up by Yih and Meek (Yih and Meek, 2007), who combine the web kernel with other simple metrics of similarity between word vectors (Dice Coefficient, Jaccard Coefficient, Overlap, Cosine, KL Divergence) in a machine learning system to provide a ranking of similar queries.

3 Proposed method

Using a search engine to collect snippets (Sahami and Heilman, 2006; Yih and Meek, 2007; Yih and Meek, 2008) takes advantage of all the optimizations performed by the retrieval engine (spelling correction, relevance scores, etc.), but it has several disadvantages: first, it is not repeatable, as the code underlying search engines is in a constant state of flux; secondly, it is usually very expensive to issue a large number of search requests; sometimes the APIs provided limit the number of requests. In this section, we describe a method which overcomes these drawbacks. The distributional methods we propose for calculating similarities between words and multiword expressions profit from the use of a large Web-based corpus.

The contextual vectors for a query can be collected by identifying the contexts in which the query appears. Queries such as *[buy a book]* and *[buy some books]* are supposed to appear close to similar context words in a bag-of-words model, and they should have a high similarity. However, there are two reasons why this would yield poor results:

First, as the length of the queries grows, the probability of finding exact queries in the corpus shrinks quickly. As an example, when issuing the queries *[Lindsay Lohan pets]* and *[Britney Spears pets]* to Google enclosed in double quotes, we obtain only 6 and 760 results, respectively. These are too few occurrences in order to collect meaningful statistics about the contexts of the queries.

Secondly, many user queries are simply a concatenation of keywords with weak or no underlying syntax. Therefore, even if they are popular queries, they may not appear as such in well-formed text found in web documents. For example, queries like *[hollywood dvd cheap]*, enclosed in double quotes, retrieve less than 10 results. Longer queries, such as *[hotel cheap new york fares]*, are still meaningful, but do not appear frequently in web documents.

In order to use of distributional similarities in the query setting, we propose the following method. Given a query of interest $p = [w_1, w_2, ..., w_n]$:

1. For each word w_i collect all words that appear close to w_i in the web corpus (i.e., a bag-fo-words models). Empirically we have chosen all the words whose distance to w_i is less or equal to 3. This gives us a vector of context words and frequencies for each of the words in the query, $\vec{v_i} = (f_{i1}, f_{i2}, ..., f_{i|V|})$, where $|V|$ is the size of the corpus vocabulary.
2. Represent the query p with a vector of words, and the weight associated to each word is the geometric mean of the frequencies for the word in the original vectors:

$$\vec{qv} = \left(\left(\prod_{i=1}^{|n|} f_{i1} \right)^{\frac{1}{n}}, \left(\prod_{i=1}^{|n|} f_{i2} \right)^{\frac{1}{n}}, ..., \left(\prod_{i=1}^{|n|} f_{i|V|} \right)^{\frac{1}{n}} \right)$$

3. Apply the χ^2 test as a weighting function test to measure whether the query and the contextual feature are conditionally independent.
4. Given two queries, use the cosine between their vectors to calculate their similarity.

The motivations for this approach are: the geometric mean is a way to approximate a boolean AND operation between the vectors, while at the same time keeping track of the magnitude of the frequencies. Therefore, if two queries only differ on a very general word, e.g. *[books]* and either *[buy books]* or *[some books]*, the vector associated to the general words (*buy* or *some* in the example) will have non-zero values for most of the contextual features, because they are not topically constrained; and the vectors for the queries will have similar sets of features with non-zero values. Equally relevant, terms that are closely related will appear in the proximity of a similar set of words and will have similar vectors. For example, if the two queries are *Sir Arthur Conan Doyle books* and *Sir Arthur Conan Doyle novels*, given that the vectors for *books* and *novels* are expected to have similar features, these two queries

Contextual word	acid	fast	bacteria	Query
acidogenicity	11	6	4	6.41506
auramin	2	5	2	2.71441
bacillae	3	10	4	4.93242
carbolfuchsin	1	28	2	8.24257
dehydrogena	5	3	3	3.55689
diphtheroid	5	9	92	16.05709
fuchsine	42	3	4	7.95811
glycosilation	3	2	3	2.62074

Table 1: Example of context words for the query *[acid fast bacteria]*.

	0	1	2	3	4
0	280	95	14	1	0
1	108	86	65	4	0
2	11	47	83	16	0
3	1	2	17	45	2
4	0	0	1	1	2

Table 2: Confusion matrix for the pairs in the goldstandard. Rows represent first rater scores, and columns second rater scores.

will receive a high similarity score.

On the other hand, this combination also helps in reducing word ambiguity. Consider the query *bank account*; the bag-of-words vector for *bank* will contain words related to the various senses of the word, but when combining it to *account* only the terms that belong to the financial domain and are shared between the two vectors will be included in the final query vector.

Finally, we note that the geometric mean provides a clean way to encode the pair-wise similarities of the individual words of the phrase. One can interpret the cosine similarity metric as the magnitude of the vector constructed by the scalar product of the individual vectors. Our approach scales this up by taking the scalar product of the vectors for all words in the phrase and then scaling them by the number of words (i.e., the geometric mean). Instead of computing the magnitude of this vector, we use it to compute similarities for the entire phrase.

As an example of the proposed procedure, Table 1 shows a random sample of the contextual features collected for the words in the query *[acid fast bacteria]*, and how the query's vector is generated by using the geometric mean of the frequencies of the features in the vectors for the query words.

4 Experiments and results

4.1 Experimental settings

To collect the contextual features for words and phrases, we have used a corpus of hundreds of millions of documents crawled from the Web in August 2008. An HTML parser is used to extract text and non-English documents are discarded. After process, the remaining corpus contains hundreds of billions of words.

As a source of keywords, we have used the top one and a half million English queries sent to the Google search engine after being fully anonymized. We have calculated the pairwise similarity between all queries, which would potentially return 2.25 trillion similarity scores, but in practice returns a much smaller number as many pairs have non-overlapping contexts.

As a baseline, we have used a new implementation of the Web Kernel similarity (Sahami and Heilman, 2006). The parameters are set the same as reported in the paper with the exception of the snippet size; in their study, the size was limited to 1,000 characters and in our system, the normal snippet returned by Google is used (around 160 characters).

In order to evaluate our system, we prepared a goldstandard set of query similarities. We have randomly sampled 65 queries from our full dataset, and obtained the top 20 suggestions from both the Sahami system and the distributional similarities system. Two human raters have rated the original query and the union of the sets of suggestions, using the same 5-point Likert scale that Sahami used. Table 2 shows the confusion matrix of scores between the two raters. Most of the disagreements are between the scores 0 and 1, which means that probably it was not clear enough whether the queries were unrelated or only slightly related. It is also noteworthy that in this case, very few rewritten queries were classified as being better than the original, which also suggests to us that probably we could remove the topmost score from the classifications scale.

We have evaluated inter-judge agreement in the following two ways: first, using the weighted Kappa score, which has a value of 0.7111. Second, by grouping the pairs judged as irrelevant or slightly relevant (scores 0 and 1) as a class containing negative examples, and the pairs judged as very relevant, equal or better (scores 2 through 4) as a class containing positive examples. Using this two-class clas-

Method	Prec@1	Prec@3	Prec@5	mAP	AUC
Web Kernel	0.39	0.35	0.32	0.49	0.22
Unigrams	0.47	0.53	0.47	0.57	0.26
N-grams	0.70	0.57	0.52	0.71	0.54

Table 3: Results. mAP is mean average precision, and AUC is the area under the precision/recall curve.

sification, Cohen's Kappa score becomes 0.6171. Both scores indicates substantial agreement amongst the raters.

The data set thus collected is a ranked list of suggestions for each query[1], and can be used to evaluate any other suggestion-ranking system.

4.2 Experiments and results

As an evolution of the distributional similarities approach, we also implemented a second version where the queries are chunked into phrases. The motivation for the second version is that, in some queries, like *[new york cheap hotel]*, it makes sense to handle *new york* as a single phrase with a single associated context vector collected from the web corpus. The list of valid n-grams is collected by combining several metrics, e.g. whether Wikipedia contains an entry with that name, or whether they appear quoted in query logs. The queries are then chunked greedily always preferring the longer n-gram from our list.

Table 3 shows the results of trying both systems on the same set of queries. The original system is the one called *Unigrams*, and the one that chunks the queries is the one called *N-grams*. The distributional similarity approaches outperform the web-based kernel on all the metrics, and chunking queries shows a good improvement over using unigrams.

5 Conclusions

This paper extends the vector-space model of distributional similarities to query-to-query similarities by combining different vectors using the geometric mean. We show that using n-grams to chunk the queries improves the results significantly. This outperforms the web-based kernel method, a state-of-the-art unsupervised query-to-query similarity technique, which is particularly relevant as the corpus-based method does not benefit automatically from search engine features.

[1]We plan to make it available to the research community.

References

S. Bordag. 2008. A Comparison of Co-occurrence and Similarity Measures as Simulations of Context. *Lecture Notes in Computer Science*, 4919:52.

J.R. Firth. 1957. A synopsis of linguistic theory 1930-1955. *Studies in Linguistic Analysis*, pages 1–32.

G. Grefenstette. 1992. Use of syntactic context to produce term association lists for text retrieval. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 89–97. ACM New York, NY, USA.

R. Jones, B. Rey, O. Madani, and W. Greiner. 2006. Generating query substitutions. In *Proceedings of the 15th international conference on World Wide Web*, pages 387–396. ACM New York, NY, USA.

Reiner Kraft and Jason Zien. 2004. Mining anchor text for query refinement. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, pages 666–674, New York, NY, USA. ACM.

D. Lin. 1998. Extracting Collocations from Text Corpora. In *First Workshop on Computational Terminology*, pages 57–63.

Stefan Riezler, Yi Liu, and Alexander Vasserman. 2008. Translating Queries into Snippets for Improved Query Expansion. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING'08)*.

E. Riloff and J. Shepherd. 1997. A corpus-based approach for building semantic lexicons. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 117–124. Association for Computational Linguistics.

M. Sahami and T.D. Heilman. 2006. A web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of the 15th international conference on World Wide Web*, pages 377–386.

G. Salton, A. Wong, and CS Yang. 1975. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620.

Egidio Terra and Charles L.A. Clarke. 2004. Scoring missing terms in information retrieval tasks. In *CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management*, pages 50–58, New York, NY, USA. ACM.

W. Yih and C. Meek. 2007. Improving Similarity Measures for Short Segments of Text. In *Proceedings of the Natural Conference on Artificial Intelligence*, volume 2, page 1489. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999.

W. Yih and C. Meek. 2008. Consistent Phrase Relevance Measures. *Data Mining and Audience Intelligence for Advertising (ADKDD 2008)*, page 37.

Text Categorization from Category Name via Lexical Reference

Libby Barak
Department of Computer Science
University of Toronto
Toronto, Canada M5S 1A4
libbyb@cs.toronto.edu

Ido Dagan and **Eyal Shnarch**
Department of Computer Science
Bar-Ilan University
Ramat-Gan 52900, Israel
{dagan, shey}@cs.biu.ac.il

Abstract

Requiring only category names as user input is a highly attractive, yet hardly explored, setting for text categorization. Earlier bootstrapping results relied on similarity in LSA space, which captures rather coarse contextual similarity. We suggest improving this scheme by identifying concrete references to the category name's meaning, obtaining a special variant of lexical expansion.

1 Introduction

Topical Text Categorization (TC), the task of classifying documents by pre-defined topics, is most commonly addressed as a supervised learning task. However, the supervised setting requires a substantial amount of manually labeled documents, which is often impractical in real-life settings.

Keyword-based TC methods (see Section 2) aim at a more practical setting. Each category is represented by a list of characteristic keywords, which should capture the category meaning. Classification is then based on measuring similarity between the category keywords and the classified documents, typically followed by a bootstrapping step. The manual effort is thus reduced to providing a keyword list per category, which was partly automated in some works through clustering.

The keyword-based approach still requires non-negligible manual work in creating a representative keyword list per category. (Gliozzo et al., 2005) succeeded eliminating this requirement by using the category name alone as the initial keyword, yet obtaining superior performance within the keyword-based approach. This was achieved by measuring similarity between category names and documents in *Latent Semantic* space (LSA), which implicitly captures contextual similarities for the category name through unsupervised dimensionality reduction. Requiring only category names as user input seems very attractive, particularly when labeled training data is too costly while modest performance (relative to supervised methods) is still useful.

The goal of our research is to further improve the scheme of text categorization from category name, which was hardly explored in prior work. When analyzing the behavior of the LSA representation of (Gliozzo et al., 2005) we noticed that it captures two types of similarities between the category name and document terms. One type regards words which refer specifically to the category name's meaning, such as *pitcher* for the category `Baseball`. However, typical context words for the category which do not necessarily imply its specific meaning, like *stadium*, also come up as similar to *baseball* in LSA space. This limits the method's precision, due to false-positive classifications of contextually-related documents that do not discuss the specific category topic (such as other sports documents wrongly classified to `Baseball`). This behavior is quite typical for query expansion methods, which expand a query with contextually correlated terms.

We propose a novel scheme that models separately these two types of similarity. For one, it identifies words that are likely to refer *specifically* to the category name's meaning (Glickman et al., 2006), based on certain relations in WordNet and

Proceedings of NAACL HLT 2009: Short Papers, pages 33–36,
Boulder, Colorado, June 2009.

Wikipedia. In tandem, we assess the general contextual fit of the category topic using an LSA model, to overcome lexical ambiguity and passing references. The evaluations show that tracing lexical references indeed increases classification precision, which in turn improves the eventual classifier obtained through bootstrapping.

2 Background: Keyword-based Text Categorization

The majority of keyword-based TC methods fit the general bootstrapping scheme outlined in Figure 1, which is cast in terms of a vector-space model. The simplest version for step 1 is manual generation of the keyword lists (McCallum and Nigam, 1999). (Ko and Seo, 2004; Liu et al., 2004) partly automated this step, using clustering to generate candidate keywords. These methods employed a standard term-space representation in step 2.

As described in Section 1, the keyword list in (Gliozzo et al., 2005) consisted of the category name alone. This was accompanied by representing the category names and documents (step 2) in LSA space, obtained through cooccurrence-based dimensionality reduction. In this space, words that tend to cooccur together, or occur in similar contexts, are represented by similar vectors. Thus, vector similarity in LSA space (in step 3) captures implicitly the similarity between the category name and contextually related words within the classified documents.

Step 3 yields an initial similarity-based classification that assigns a single (most similar) category to each document, with $Sim(c, d)$ typically being the cosine between the corresponding vectors. This classification is used, in the subsequent bootstrapping step, to train a standard supervised classifier (either single- or multi-class), yielding the eventual classifier for the category set.

3 Integrating Reference and Context

Our goal is to augment the coarse contextual similarity measurement in earlier work with the identification of concrete references to the category name's meaning. We were mostly inspired by (Glickman et al., 2006), which coined the term *lexical reference* to denote concrete references in text to the specific meaning of a given term. They further showed that

Input: set of categories and unlabeled documents **Output:** a classifier
1. Acquire a keyword list per category 2. Represent each category c and document d as vectors in a common space 3. For each document d $Cat_{Sim}(d) = argmax_c(Sim(c, d))$ 4. Train a supervised classifier on step (3) output

Figure 1: Keyword-based categorization scheme

Category name	WordNet	Wikipedia
`Cryptography`	*decipher*	*digital signature*
`Medicine`	*cardiology*	*biofeedback, homeopathy*
`Macintosh`		*Apple Mac, Mac*
`Motorcycle`	*bike, cycle*	*Honda XR600*

Table 1: Referring terms from WordNet and Wikipedia

an entailing text (in the textual entailment setting) typically includes a concrete reference to each term in the entailed statement. Analogously, we assume that a relevant document for a category typically includes concrete terms that refer *specifically* to the category name's meaning.

We thus extend the scheme in Figure 1 by creating two vectors per category (in steps 1 and 2): a *reference vector* $\vec{c}_{ref}$ in term space, consisting of referring terms for the category name; and a *context vector* $\vec{c}_{con}$, representing the category name in LSA space, as in (Gliozzo et al., 2005). Step 3 then computes a combined similarity score for categories and documents based on the two vectors.

3.1 References to category names

Referring terms are collected from WordNet and Wikipedia, by utilizing relations that are likely to correspond to lexical reference. Table 1 illustrates that WordNet provides mostly referring terms of general terminology while Wikipedia provides more specific terms. While these resources were used previously for text categorization, it was mostly for enhancing document representation in supervised settings, e.g. (Rodríguez et al., 2000).

WordNet. Referring terms were found in WordNet starting from relevant senses of the category name and transitively following relation types that correspond to lexical reference. To that end, we

specified for each category name those senses which fit the category's meaning, such as the *outer space* sense for the category `Space`.[1]

A category name sense is first expanded by its synonyms and derivations, all of which are then expanded by their hyponyms. When a term has no hyponyms it is expanded by its meronyms instead, since we observed that in such cases they often specify unique components that imply the holonym's meaning, such as *Egypt* for *Middle East*. However, when a term is not a leaf in the hyponymy hierarchy then its meronyms often refer to generic sub-parts, such as *door* for *car*. Finally, the hyponyms and meronyms are expanded by their derivations. As a common heuristic, we considered only the most frequent senses (top 4) of referring terms, avoiding low-ranked (rare) senses which are likely to introduce noise.

Wikipedia. We utilized a subset of a lexical reference resource extracted from Wikipedia (anonymous reference). For each category name we extracted referring terms of two types, capturing hyponyms and synonyms. Terms of the first type are Wikipedia page titles for which the first definition sentence includes a syntactic "is-a" pattern whose complement is the category name, such as *Chevrolet* for the category `Autos`. Terms of the second type are extracted from Wikipedia's redirect links, which capture synonyms such as *x11* for `Windows-X`.

The reference vector $\vec{c}_{ref}$ for a category consists of the category name and all its referring terms, equally weighted. The corresponding similarity function is $Sim_{ref}(c,d) = \cos(\vec{c}_{ref}, \vec{d}_{term}))$, where $\vec{d}_{term}$ is the document vector in term space.

3.2 Incorporating context similarity

Our key motivation is to utilize Sim_{ref} as the basis for classification in step 3 (Figure 1). However, this may yield false positive classifications in two cases: (a) inappropriate sense of an ambiguous referring term, e.g., the narcotic sense of *drug* should not yield classification to `Medicine`; (b) a passing reference, e.g., an analogy to *cars* in a software document, should not yield classification to `Autos`.

In both these cases the overall context in the document is expected to be atypical for the triggered category. We therefore measure the contextual similarity between a category c and a document d utilizing LSA space, replicating the method in (Gliozzo et al., 2005): $\vec{c}_{con}$ and $\vec{d}_{LSA}$ are taken as the LSA vectors of the category name and the document, respectively, yielding $Sim_{con}(c,d) = \cos(\vec{c}_{con}, \vec{d}_{LSA}))$.[2]

The overall similarity score of step 3 is defined as $Sim(c,d) = Sim_{ref}(c,d) \cdot Sim_{con}(c,d)$. This formula fulfils the requirement of finding at least one referring term in the document; otherwise $Sim_{ref}(c,d)$ would be zero. $Sim_{con}(c,d)$ is computed in the reduced LSA space and is thus practically non-zero, and would downgrade $Sim(c,d)$ when there is low contextual similarity between the category name and the document. Documents for which $Sim(c,d) = 0$ for all categories are omitted.

4 Results and Conclusions

We tested our method on the two corpora used in (Gliozzo et al., 2005): 20-NewsGroups, classified by a single-class scheme (single category per document), and Reuters-10 [3], of a multi-class scheme. As in their work, non-standard category names were adjusted, such as *Foreign exchange* for `Money-fx`.

4.1 Initial classification

Table 2 presents the results of the initial classification (step 3). The first 4 lines refer to classification based on Sim_{ref} alone. As a baseline, including only the category name in the reference vector (*CatName*) yields particularly low recall. Expansion by *WordNet* is notably more powerful than by the automatically extracted *Wikipedia* resource; still, the latter consistently provides a small marginal improvement when using both resources (*Reference*), indicating their complementary nature.

As we hypothesized, the *Reference* model achieves much better precision than the *Context* model from (Gliozzo et al., 2005) alone (Sim_{con}). For 20-NewsGroups the recall of *Reference* is limited, due to partial coverage of our current expansion

[1] We assume that it is reasonable to specify relevant senses as part of the typically manual process of defining the set of categories and their names. Otherwise, when expanding names through all their senses F1-score dropped by about 2%.

[2] The original method includes a Gaussian Mixture rescaling step for Sim_{con}, which wasn't found helpful when combined with Sim_{ref} (as specified next).

[3] 10 most frequent categories in *Reuters-21578*

	Reuters-10			20 Newsgroups		
Method	R	P	F1	R	P	F1
CatName	0.22	0.67	0.33	0.19	0.55	0.28
WordNet	0.67	0.78	0.72	0.29	0.56	0.38
Wikipedia	0.24	0.68	0.35	0.22	0.57	0.31
Reference	0.69	0.80	0.74	0.31	0.57	0.40
Context	0.59	0.64	0.61	**0.46**	0.46	**0.46**
Combined	**0.71**	**0.82**	**0.76**	0.32	**0.58**	0.41

Table 2: Initial categorization results (step 3)

Method	Feature Set	Reuters-10			20 NG
		R	P	F1	F1
Reference	TF-IDF	0.91	0.50	0.65	0.51
	LSA	0.89	0.67	0.76	**0.56**
Context	TF-IDF	0.84	0.48	0.61	0.48
	LSA	0.73	0.56	0.63	0.44
Combined	TF-IDF	**0.92**	0.50	0.65	0.52
	LSA	0.89	**0.71**	**0.79**	**0.56**

Table 3: Final bootstrapping results (step 4)

resources, yielding a lower F1. Yet, its higher precision pays off for the bootstrapping step (Section 4.2). Finally, when the two models are *Combined* a small precision improvement is observed.

4.2 Final bootstrapping results

The output of step 3 was fed as standard training for a binary SVM classifier for each category (step 4). We used the default setting for SVM-light, apart from the *j* parameter which was set to the number of categories in each data set, as suggested by (Morik et al., 1999). For Reuters-10, classification was determined independently by the classifier of each category, allowing multiple classes per document. For 20-NewsGroups, the category which yielded the highest classification score was chosen (one-versus-all), fitting the single-class setting. We experimented with two document representations for the supervised step: either as vectors in tf-idf weighted term space or as vectors in LSA space.

Table 3 shows the final classification results.[4] First, we observe that for the noisy bootstrapping training data LSA document representation is usually preferred. Most importantly, our *Reference* and *Combined* models clearly improve over the earlier *Context*. Combining reference and context yields some improvement for Reuters-10, but not for 20-NewsGroups. We noticed though that the actual accuracy of our method on 20-NewsGroups is notably higher than measured relative to the gold standard, due to its single-class scheme: in many cases, a document should truly belong to more than one category while that chosen by our algorithm was counted as false positive. Future research is proposed to increase the method's recall via broader coverage lexical reference resources, and to improve its precision through better context models than LSA, which was found rather noisy for quite a few categories.

To conclude, the results support our main contribution – the benefit of identifying *referring terms* for the category name over using noisier context models alone. Overall, our work highlights the potential of text categorization from category names when labeled training sets are not available, and indicates important directions for further research.

[4]Notice that P=R=F1 when *all* documents are classified to a single class, as in step 4 for 20-NewsGroups, while in step 3 some documents are not classified, yielding distinct P/R/F1.

Acknowledgments

The authors would like to thank Carlo Strapparava and Alfio Gliozzo for valuable discussions. This work was partially supported by the NEGEV project (www.negev-initiative.org).

References

O. Glickman, E. Shnarch, and I. Dagan. 2006. Lexical reference: a semantic matching subtask. In *EMNLP*.

A. Gliozzo, C. Strapparava, and I. Dagan. 2005. Investigating unsupervised learning for text categorization bootstrapping. In *Proc. of HLT/EMNLP*.

Y. Ko and J. Seo. 2004. Learning with unlabeled data for text categorization using bootstrapping and feature projection techniques. In *Proc. of ACL*.

B. Liu, X. Li, W. S. Lee, and P. S. Yu. 2004. Text classification by labeling words. In *Proc. of AAAI*.

A. McCallum and K. Nigam. 1999. Text classification by bootstrapping with keywords, EM and shrinkage. In *ACL Workshop for Unsupervised Learning in NLP*.

K. Morik, P. Brockhausen, and T. Joachims. 1999. Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In *Proc. of the 16th Int'l Conf. on Machine Learning*.

M. d. B. Rodríguez, J. M. Gómez-Hidalgo, and B. Díaz-Agudo, 2000. *Using WordNet to complement training information in text categorization*, volume 189 of Current Issues in Linguistic Theory, pages 353–364.

Identifying Types of Claims in Online Customer Reviews

Shilpa Arora, Mahesh Joshi and **Carolyn P. Rosé**
Language Technologies Institute
School of Computer Science
Carnegie Mellon University, Pittsburgh PA 15213
{shilpaa,maheshj,cprose}@cs.cmu.edu

Abstract

In this paper we present a novel approach to categorizing comments in online reviews as either a *qualified claim* or a *bald claim*. We argue that this distinction is important based on a study of customer behavior in making purchasing decisions using online reviews. We present results of a supervised algorithm for learning this distinction. The two types of claims are expressed differently in language and we show that syntactic features capture this difference, yielding improvement over a bag-of-words baseline.

1 Introduction

There has been tremendous recent interest in opinion mining from online product reviews and it's effect on customer purchasing behavior. In this work, we present a novel alternative categorization of comments in online reviews as either being *qualified claims* or *bald claims*.

Comments in a review are claims that reviewers make about the products they purchase. A customer reads the reviews to help him/her make a purchasing decision. However, comments are often open to interpretation. For example, a simple comment like *this camera is small* is open to interpretation until qualified by more information about whether it is small in general (for example, based on a poll from a collection of people), or whether it is small compared to some other object. We call such claims *bald claims*. Customers hesitate to rely on such bald claims unless they identify (from the context or otherwise) themselves to be in a situation similar to the customer who posted the comment. The other category of claims that are not bald are *qualified claims*. Qualified claims such as *it is small enough to fit easily in a coat pocket or purse* are more precise claims as they give the reader more details, and are less open to interpretation. Our notion of qualified claims is similar to that proposed in the argumentation literature by Toulmin (1958). This distinction of qualified vs. bald claims can be used to filter out bald claims that can't be verified. For the qualified claims, the qualifier can be used in personalizing what is presented to the reader.

The main contributions of this work are: (i) an annotation scheme that distinguishes qualified claims from bald claims in online reviews, and (ii) a supervised machine learning approach that uses syntactic features to learn this distinction. In the remainder of the paper, we first motivate our work based on a customer behavior study. We then describe the proposed annotation scheme, followed by our supervised learning approach. We conclude the paper with a discussion of our results.

2 Customer Behavior Study

In order to study how online product reviews are used to make purchasing decisions, we conducted a user study. The study involved 16 pair of graduate students. In each pair there was a customer and an observer. The goal of the customer was to decide which camera he/she would purchase using a camera review blog[1] to inform his/her decision. As the customer read through the reviews, he/she was

[1] http://www.retrevo.com/s/camera

Proceedings of NAACL HLT 2009: Short Papers, pages 37–40,
Boulder, Colorado, June 2009.

asked to think aloud and the observer recorded their observations.

The website used for this study had two types of reviews: expert and user reviews. There were mixed opinions about which type of reviews people wanted to read. About six customers could relate more with user reviews as they felt expert reviews were more like a 'sales pitch'. On the other hand, about five people were interested in only expert reviews as they believed them to be more practical and well reasoned.

From this study, it was clear that the customers were sensitive to whether a claim was qualified or not. About 50% of the customers were concerned about the reliability of the comments and whether it applied to them. Half of them felt it was hard to comprehend whether the user criticizing a feature was doing so out of personal bias or if it represented a real concern applicable to everyone. The other half liked to see comments backed up with facts or explanations, to judge if the claim could be qualified. Two customers expressed interest in comments from users similar to themselves as they felt they could base their decision on such comments more reliably. Also, exaggerations in reviews were deemed untrustworthy by at least three customers.

3 Annotation Scheme

We now present the guidelines we used to distinguish bald claims from qualified claims. A claim is called qualified if its validity or scope is limited by making the conditions of its applicability more explicit. It could be either a fact or a statement that is well-defined and attributed to some source. For example, the following comments from our data are qualified claims according to our definition,

1. *The camera comes with a lexar 16mb starter card, which stores about 10 images in fine mode at the highest resolution.*
2. *I sent my camera to nikon for servicing, took them a whole 6 weeks to diagnose the problem.*
3. *I find this to be a great feature.*

The first example is a fact about the camera. The second example is a report of an event. The third example is a self-attributed opinion of the reviewer.

Bald claims on the other hand are non-factual claims that are open to interpretation and thus cannot be verified. A straightforward example of the distinction between a bald claim and a qualified claim is a comment like *the new flavor of peanut butter is being well appreciated* vs. *from a survey conducted among 20 people, 80% of the people liked the new flavor of peanut butter.* We now present some examples of bald claims. A more detailed explanation is provided in the annotation manual[2]:

- **Not quantifiable gradable**[3] words such as *good*, *better*, *best* etc. usually make a claim bald, as there is no qualified definition of being good or better.
- **Quantifiable gradable** words such as *small*, *hot* etc. make a claim bald when used without any frame of reference. For example, a comment *this desk is small* is a bald claim whereas *this desk is smaller than what I had earlier* is a qualified claim, since the comparative *smaller* can be verified by observation or actual measurement, but whether something is *small* in general is open to interpretation.
- **Unattributed opinion or belief:** A comment that implicitly expresses an opinion or belief without qualifying it with an explicit attribution is a bald claim. For example, *Expectation is that camera automatically figures out when to use the flash*.
- **Exaggerations:** Exaggerations such as *on every visit, the food has blown us away* do not have a well defined scope and hence are not well qualified.

The two categories for gradable words defined above are similar to what Chen (2008) describes as *vagueness, non-objective measurability and imprecision*.

4 Related work

Initial work by Hu and Liu (2004) on the product review data that we have used in this paper focuses on the task of opinion mining. They propose an approach to summarize product reviews by identifying opinionated statements about the features of a product. In our annotation scheme however, we classify

[2] www.cs.cmu.edu/~shilpaa/datasets/opinion-claims/qbclaims-manual-v1.0.pdf

[3] http://en.wikipedia.org/wiki/English_grammar#Semantic_gradability

all claims in a review, not restricting to comments with feature mentions alone.

Our task is related to opinion mining, but with a specific focus on categorizing statements as either bald claims that are open to interpretation and may not apply to a wide customer base, versus qualified claims that limit their scope by making some assumptions explicit. Research in analyzing subjectivity of text by Wiebe et al. (2005) involves identifying expression of private states that cannot be objectively verified (and are therefore open to interpretation). However, our task differs from subjectivity analysis, since both bald as well as qualified claims can involve subjective language. Specifically, objective statements are always categorized as qualified claims, but subjective statements can be either bald or qualified claims. Work by Kim and Hovy (2006) involves extracting pros and cons from customer reviews and as in the case of our task, these pros and cons can be either subjective or objective.

In supervised machine learning approaches to opinion mining, the results using longer n-grams and syntactic knowledge as features have been both positive as well as negative (Gamon, 2004; Dave et al., 2003). In our work, we show that the qualified vs. bald claims distinction can benefit from using syntactic features.

5 Data and Annotation Procedure

We applied our annotation scheme to the product review dataset[4] released by Hu and Liu (2004). We annotated the data for 3 out of 5 products. Each comment in the review is evaluated as being qualified or bald claim. The data has been made available for research purposes[5].

The data was completely double coded such that each review comment received a code from the two annotators. For a total of $1,252$ review comments, the Cohen's kappa (Cohen, 1960) agreement was 0.465. On a separate dataset (365 review comments)[6], we evaluated our agreement after removing the borderline cases (only about 14%) and there was a statistically significant improvement in kappa to 0.532. Since the agreement was low, we resolved our conflict by consensus coding on the data that was used for supervised learning experiments.

6 Experiments and Results

For our supervised machine learning experiments on automatic classification of comments as qualified or bald, we used the Support Vector Machine classifier in the MinorThird toolkit (Cohen, 2004) with the default linear kernel. We report average classification accuracy and average Cohen's Kappa using 10-fold cross-validation.

6.1 Features

We experimented with several different features including standard lexical features such as word unigrams and bigrams; pseudo-syntactic features such as Part-of-Speech bigrams and syntactic features such as dependency triples[7]. Finally, we also used syntactic scope relationships computed using the dependency triples. Use of features based on syntactic scope is motivated by the difference in how qualified and bald claims are expressed in language. We expect these features to capture the presence or absence of qualifiers for a stated claim. For example, "*I didn't like this camera, but I suspect it will be a great camera for first timers.*" is a qualified claim, whereas a comment like "*It will be a great camera for first timers.*" is not a qualified claim. Analysis of the syntactic parse of the two comments shows that in the first comment the word "great" is in the scope of "suspect", whereas this is not the case for the second comment. We believe such distinctions can be helpful for our task.

We compute an approximation to the syntactic scope using dependency parse relations. Given the set of dependency relations of the form ≪relation, headWord, dependentWord≫, such as ≪AMOD,camera,great≫, an in-scope feature is defined as INSCOPE_headWord_dependentWord (INSCOPE_camera_great). We then compute a transitive closure of such in-scope features, similar to Bikel and Castelli (2008). For each in-scope feature in the entire training fold, we also create a corre-

[4] http://www.cs.uic.edu/~liub/FBS/CustomerReviewData.zip

[5] www.cs.cmu.edu/~shilpaa/datasets/opinion-claims/qbclaims-v1.0.tar.gz

[6] These are also from the Hu and Liu (2004) dataset, but not included in our dataset yet.

[7] We use the Stanford Part-of-Speech tagger and parser respectively.

Features	QBCLAIM	HL-OP
Majority	.694(.000)	.531(.000)
Unigrams	.706(.310)	.683(.359)
+Bigrams	.709(.321)	.693(.378)
+POS-Bigrams	**.726*(.353*)**	.683(.361)
+Dep-Triples	.711(.337)	.692(.376)
+In-scope	.706(.340)	.688(.367)
+Not-in-scope	.726(**.360***)	.687(.370)
+All-scope	.721(.348)	.699(.396)

Table 1: The table shows accuracy (& Cohen's kappa in parentheses) averaged across ten folds. Each feature set is individually added to the baseline set of unigram features. * - Result is marginally significantly better than unigrams-only ($p < 0.10$, using a two-sided pairwise T-test). HL-OP - Opinion annotations from Hu and Liu (2004). QBCLAIM - Qualified/Bald Claim.

sponding not-in-scope feature which triggers when either (i) the dependent word appears in a comment, but not in the transitive-closured scope of the head word, or (ii) the head word is not contained in the comment but the dependent word is present.

We evaluate the benefit of each type of feature by adding them individually to the baseline set of unigram features. Table 1 presents the results. We use the majority classifier and unigrams-only performance as our baselines. We also experimented with using the same feature combinations to learn the *opinion* category as defined by Hu and Liu (2004) [HL-OP] on the same subset of data.

It can be seen from Table 1 that using purely unigram features, the accuracy obtained is not any better than the majority classifier for qualified vs. bald distinction. However, the Part-of-Speech bigram features and the not-in-scope features achieve a marginally significant improvement over the unigrams-only baseline.

For the opinion dimension from Hu and Liu (2004), there was no significant improvement from the type of syntactic features we experimented with. Hu and Liu (2004)'s opinion category covers several different types of opinions and hence finer linguistic distinctions that help in distinguishing qualified claims from bald claims may not apply in that case.

7 Conclusions

In this work, we presented a novel approach to review mining by treating comments in reviews as claims that are either qualified or bald. We argued with examples and results from a user study as to why this distinction is important. We also proposed and motivated the use of syntactic scope as an additional type of syntactic feature, apart from those already used in opinion mining literature. Our evaluation demonstrates a marginally significant positive effect of a feature space that includes these and other syntactic features over the purely unigram-based feature space.

Acknowledgments

We would like to thank Dr. Eric Nyberg for the helpful discussions and the user interface for doing the annotations. We would also like to thank all the anonymous reviewers for their helpful comments.

References

Daniel Bikel and Vittorio Castelli. *Event Matching Using the Transitive Closure of Dependency Relations*. Proceedings of ACL-08: HLT, Short Papers, pp. 145–148.

Wei Chen. 2008. *Dimensions of Subjectivity in Natural Language*. In Proceedings of ACL-HLT'08. Columbus Ohio.

Jacob Cohen. 1960. *A Coefficient of Agreement for Nominal Scales*. Educational and Psychological Measurement, Vol. 20, No. 1., pp. 37-46.

William Cohen. 2004. *Minorthird: Methods for Identifying Names and Ontological Relations in Text using Heuristics for Inducing Regularities from Data*. http://minorthird.sourceforge.net/

Kushal Dave, Steve Lawrence and David M. Pennock 2006. *Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews*. In Proc of WWW'03.

Michael Gamon. *Sentiment classification on customer feedback data: noisy data, large feature vectors, and the role of linguistic analysis*. Proceedings of the International Conference on Computational Linguistics (COLING).

Minqing Hu and Bing Liu. 2004. *Mining and Summarizing Customer Reviews*. In Proc. of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.

Soo-Min Kim and Eduard Hovy. 2006. *Automatic Identification of Pro and Con Reasons in Online Reviews*. In Proc. of the COLING/ACL Main Conference Poster Sessions.

Stephen Toulmin 1958 *The Uses of Argument*. Cambridge University Press.

Janyce Wiebe, Theresa Wilson, and Claire Cardie 2005. *Annotating expressions of opinions and emotions in language*. Language Resources and Evaluation, volume 39, issue 2-3, pp. 165-210.

Towards Automatic Image Region Annotation - Image Region Textual Coreference Resolution

Emilia Apostolova
College of Computing and Digital Media
DePaul University
Chicago, IL 60604, USA
emilia.aposto@gmail.com

Dina Demner-Fushman
Communications Engineering Branch
National Library of Medicine
Bethesda, MD 20894, USA
ddemner@mail.nih.gov

Abstract

Detailed image annotation necessary for reliable image retrieval involves not only annotating the image as a single artifact, but also annotating specific objects or regions within the image. Such detailed annotation is a costly endeavor and the available annotated image data are quite limited. This paper explores the feasibility of using image captions from scientific journals for the purpose of automatically annotating image regions. Salient image clues, such as an object location within the image or an object color, together with the associated explicit object mention, are extracted and classified using rule-based and SVM learners.

1 Introduction

The profusion of digitally available images has naturally led to an interest in the field of automatic image annotation and retrieval. A number of studies attempt to associate image regions with the corresponding concepts. In (Duygulu et al., 2002), for example, the problem of annotation is treated as a translation from a set of image segments (or blobs) to a set of words. Modeling the association between blobs and words for the purpose of automated annotation has also been proposed by (Barnard et al., 2003; Jeon et al., 2003).

A recurring hindrance that appears in studies aiming at automatic image region annotation is the lack of an appropriate dataset. All of the above studies use the Corel image dataset that consists of 60,000 images annotated with 3 to 5 keywords. The need for an image dataset with annotated image regions has been recognized by many researchers. For example, Russell et al (2008) have developed a tool and a general purpose image database designed to delineate and annotate objects within image scenes.

The need for an image dataset with annotated object boundaries appears to be especially pertinent in the biomedical field. Organizing and using for research the available medical imaging data proved to be a challenge and a goal of the ongoing research. Rubin et al (2008), for example, propose an ontology and annotation tool for semantic annotation of image regions in radiology.

However, creating a dataset of image regions manually annotated and delineated by domain experts, is a costly enterprise. Any attempts to automate or semi-automate the process would be of a substantial value.

This work proposes an approach towards automatic annotation of regions of interest in images used in scientific publications. Publications abundant in image data are an untapped source of annotated image data. Due to publication standards, meaningful image captions are almost always provided within scientific articles. In addition, image Regions of Interest (ROIs) are commonly referred to within the image caption. Such ROIs are also commonly delineated with some kind of an overlay that helps locating the ROI. This is especially true for hard to interpret scientific images such as radiology images. ROIs are also described in terms of location within the image, or by the presence of a particular color. Identifying ROI mentions within image captions and visual clues pinpointing the ROI within the image would be the first step in building an object

Proceedings of NAACL HLT 2009: Short Papers, pages 41–44,
Boulder, Colorado, June 2009.

1. Object Location - explicit ROI location, e.g. front row, background, top, bottom, left, right. *Shells of planktonic animals called formainifera record climatic conditions as they are formed. This one, Globigerinoides ruber, lives year-round at the surface of the Sargasso Sea. The form of the live animal is shown **at right**, and its shell, which is actually about the size of a fine grain of sand, **at left**.*
2. Object Color - presence of a distinct color that identifies a ROI. *Anterior SSD image shows an elongated splenorenal varix (**blue area**). The varix travels from the splenic hilar region inferiorly along the left flank, down into the pelvis, and eventually back up to the left renal vein via the left gonadal vein. The kidney is encoded **yellow**, the portal system is encoded **magenta**, and the spleen is encoded **tan**.*
3. Overlay Marker - an overlay marker used to pinpoint the location of the ROI, e.g. arrows, asterisks, bounding boxes, or circles. *Transverse sonograms obtained with a 7.5-MHz linear transducer in the subareolar region. **The straight arrows** show a dilated tubular structure. **The curved arrow** indicates an intraluminal solid mass.*
4. Overlay Label - an overlay label used to pinpoint the location of the ROI, e.g. numbers, letters, words, abbreviations. *Location of the calf veins. Transverse US image just above ankle demonstrates the paired posterior tibial veins (**V**) and posterior tibial artery (**A**) imaged from a posteromedial approach. Note there is inadequate venous flow velocity to visualize with color Doppler without flow augmentation.*

Table 1: Image Markers divided into four categories, followed by a sample image caption[1] in which Image Markers are marked in bold, Image Marker Referents are underlined.

delineated and annotated image dataset.

2 Problem Definition

The goal of this research is to locate visually salient image region characteristics in the text surrounding scientific images that could be used to facilitate the delineation of the image object boundaries. This task could be broken down into two related subtasks - 1) locating and classifying textual clues for visually salient ROI features (Image Markers), and 2) locating the corresponding ROI text mentions (Image Marker Referents). Table 1 gives a classification of Image Markers including examples of Image Markers and Image Marker Referents. Figure 1 shows the frequency of Image Marker occurrences.

[1]The captions were extracted from Radiology and Radiographics © Radiological Society of North America and Oceanus © Woods Hole Oceanographic Institution.

3 Related Work

Cohen et al (2003) attempt to identify what they refer to as "image pointers" within captions in biomedical publications. The image pointers of interest are, for example, image panel labels, or letters and abbreviations used as an overlay within the image, similar to the Overlay Labels described in Table 1. They developed a set of hand-crafted rules, and a learning method involving Boosted Wrapper Induction on a dataset consisting of biomedical articles related to fluorescence microscope images.

Deschacht and Moens (2007) analyze text surrounding images in news articles trying to identify persons and objects in the text that appear in the corresponding image. They start by extracting persons' names and visual objects using Named Entity Recognition (NER) tools. Next, they measure the "salience" of the extracted named entities within the text with the assumption that more salient named entities in the text will also be present in the accompanying image.

Davis et al (2003) develop a NER tool to identify references to a single art object (for example a specific building within an image) in text related to art images for the purpose of automatic cataloging of images. They take a semi-supervised approach to locating the named entities of interest by first providing an authoritative list of art objects of interest and then seeking to match variants of the seed named entities in related text.

4 Experimental Methods and Results

4.1 Dataset

The chosen dateset contains more than 60,000 images together with their associated captions from three online life and earth sciences journals[1]. 400 randomly selected image captions were manually annotated by a single annotator with their Image Markers and Image Marker Referents and used for testing and for cross-validation respectively

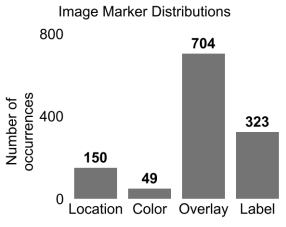


Figure 1: Distribution of Image Marker types across 400 annotated image captions.

in the two methods described below.

4.2 Rule Based Approach

First, we developed a two-stage rule-based, bootstrapping algorithm for locating the image markers and their coreferents from unannotated data. The algorithm is based on the observation that textual image markers commonly appear in parentheses and are usually closely related semantic concepts. Thus the seed for the algorithm consists of:

1. The predominant syntactic pattern - parentheses, as in *'hooking of the soft palate (arrow)'*. This pattern could easily be captured by a regular expression and doesn't require sentence parsing.

2. A dozen seed phrases (e.g *'left'*, *'circle'*, *'asterisk'*, *'blue'*) identified by initially annotating a small subset of the data (20 captions). Wordnet was used to look up and prepare a list of their corresponding inherited hypernyms. This hypernym list contains concepts such as *'a spatially limited location'*, *'a two-dimensional shape'*, *'a written or printed symbol'*, *'a visual attribute of things that results from the light they emit or transmit or reflect'*. Best results were achieved when inherited hypernyms up to the third parent were used.

In the first stage of the algorithm, all image captions were searched for parenthesized expressions that share the seed hypernyms. This step of the algorithm will result in high precision, but a low recall since image markers do not necessarily appear in parentheses. To increase recall, in stage 2 a full text search was performed for the stemmed versions of the expressions identified in stage 1.

A baseline measure was also computed for the identification of the Image Marker Referents using a simple heuristic - the coreferent of the Image Marker is usually the closest Noun Phrase (NP). In the case of parenthesized image markers, it is the closest NP to the left of the image marker; in the case of non-parenthesized image markers, the referent is usually the complement of the verb; and in the case of passive voice, the NP preceding the verb phrase. The Stanford parser was used to parse the sentences.

Table 2 summarizes the results validated against the annotated dataset (excluding the 20 captions used to identify the seed phrases). It appears that the relatively low accuracy for Image Marker Referent identification was mostly due to parsing errors since the syntactic structure of the image caption texts is quite distinct from the Penn Treebank dataset used for training the Stanford parser.

	Precision	Recall	F1-score
Image Marker	87.70	68.10	76.66
Image Marker Referent	Accuracy	59.10	

Table 2: Rule-based approach results for Image Marker and Image Marker Referent identification. Image Marker Referent results are reported as accuracy because the algorithm involves locating an Image Marker Referent for each Image Marker. Referent identification accuracy was computed for all annotated Image Markers.

Kind	k_{-5}	...	k_0	...	k_{+5}
Orth	o_{-5}	...	o_0	...	o_{+5}
Stem	s_{-5}	...	s_0	...	s_{+5}
Hypernym	h_{-5}	...	h_0	...	h_{+5}
Dep Path	d_{-5}	...	d_0	...	d_{+5}
Category			$\mathbf{[c_0]}$		

Table 3: Features from a surrounding token window are used to classify the current token into category $\mathbf{[c_0]}$. Best results were achieved with a five-token window.

4.3 Support Vector Machines

Next we explored the possibility of improving the rule-based method results by applying a machine learning technique on the set of annotated data. Support Vector Machines (SVM) (Vapnik, 2000) was the approach taken because it is a state-of-the-art classification approach proven to perform well on many NLP tasks.

In our approach, each sentence was tokenized, and tokens were classified as Beginning, Inside, or Outside an Image Marker type or Image Marker Referent. Image Marker Referents are not related to Image Markers and creating a classifier trained on this task is planned as future work. SVM classifiers were trained for each of these categories, and combined via 'one-vs-all' classification (the category of the classifier with the largest output was selected). Features of the surrounding context are used as shown in Table 3 and Table 4.

Table 5 summarizes the results of a 10-fold cross-validation. SVM performed well overall for identifying Image Markers, Location being the hardest because of higher variability of expressing ROI position. Image Marker Referents are harder to classify,

Token Kind	The general type of the sentence token (Word, Number, Symbol, Punctuation, White space).
Orthography	Orthographic categorization of the token (Upper initial, All capitals, Lower case, Mixed case).
Stem	The stem of the token, extracted with the Porter stemmer.
Wordnet Superclass	Wordnet hypernyms (nouns, verbs); the hypernym of the derivationally related form (adjectives); the superclass of the pertanym (adverbs).
POS Category	POS categories extracted using Brill's tagger.
Dependency Path*	The smallest sentence parse subtree including both the current token and the annotated image marker(s), encoded as an undirected path across POS categories.

Table 4: Orthographic, semantic, and grammatical classification features computed for each token (*Dependency Path is used only for classifying Image Marker Referents).

as deeper syntactic knowledge is necessary. Idiosyncratic syntactic structures in image captions pose a problem for the general-purpose trained Stanford parser and performance is hindered by the accuracy of computing Dependency Path feature.

5 Conclusion and Future Work

We explored the feasibility of determining the content of ROIs in images from scientific publications using image captions. We developed a two-stage rule-based approach that utilizes WordNet to find ROI pointers (Image Markers) and their referents. We also explored a supervised machine learning approach. Both approaches are promising. The rule-based approach seeded with a small manually annotated set resulted in 78.7% precision and 68.1% recall for Image Markers recognition. The SVM approach (which requires a greater annotation effort) outperformed the rule based approach (p=93.6%, r=87.7%). Future plans include training SVMs on the results of the rule-based annotation. Further work is also needed in improving Image Marker Referent identification and co-reference resolution. We also plan to involve two annotators in order to collect a more robust dataset based on inter-annotator agreement.

	Precision	Recall	F1-score
Location	60.93	45.15	51.86
Color	100.00	51.32	67.82
Overlay Marker	97.43	95.39	96.39
Overlay Label	85.74	87.69	86.70
Overall	93.64	87.69	90.56
Image Marker Referent	Accuracy	61.15	

Table 5: SVM classification results for the four types of Image Markers, and for Image Marker Referents. LibSVM software was used (3-degree polynomial kernel, cost parameter = 1, τ = 0.6 empirically determined).

References

K. Barnard, P. Duygulu, D. Forsyth, N. de Freitas, D.M. Blei, and M.I. Jordan. 2003. Matching words and pictures. *The Journal of Machine Learning Research*, 3:1107–1135.

W.W. Cohen, R. Wang, and R.F. Murphy. 2003. Understanding captions in biomedical publications. In *Proceedings of the 9th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 499–504. ACM New York, NY, USA.

P.T. Davis, D.K. Elson, and J.L. Klavans. 2003. Methods for precise named entity matching in digital collections. In *Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries*, pages 125–127. IEEE Computer Society Washington, DC, USA.

K. Deschacht and M. Moens. 2007. Text analysis for automatic image annotation. In *Proceedings of the 45th Annual ACL Meeting*, pages 1000–1007. ACL.

P. Duygulu, K. Barnard, JFG de Freitas, and D.A. Forsyth. 2002. Object Recognition as Machine Translation: Learning a Lexicon for a Fixed Image Vocabulary. *LECTURE NOTES IN COMPUTER SCIENCE*, pages 97–112.

J. Jeon, V. Lavrenko, and R. Manmatha. 2003. Automatic image annotation and retrieval using cross-media relevance models. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 119–126. ACM New York, NY, USA.

D. Rubin, P. Mongkolwat, V. Kleper, K. Supekar, and D. Channin. 2008. Medical imaging on the Semantic Web: Annotation and image markup. In *AAAI Spring Symposium Series, Semantic Scientific Knowledge Integration*.

B.C. Russell, A. Torralba, K.P. Murphy, and W.T. Freeman. 2008. LabelMe: A Database and Web-Based Tool for Image Annotation. *International Journal of Computer Vision*, 77(1):157–173.

V.N. Vapnik. 2000. *The Nature of Statistical Learning Theory*. Springer.

TESLA: A Tool for Annotating Geospatial Language Corpora

Nate Blaylock and **Bradley Swain** and **James Allen**
Institute for Human and Machine Cognition (IHMC)
Pensacola, Florida, USA
{blaylock,bswain,jallen}@ihmc.us

Abstract

In this paper, we present The gEoSpatial Language Annotator (TESLA)—a tool which supports human annotation of geospatial language corpora. TESLA interfaces with a GIS database for annotating grounded geospatial entities and uses Google Earth for visualization of both entity search results and evolving object and speaker position from GPS tracks. We also discuss a current annotation effort using TESLA to annotate location descriptions in a geospatial language corpus.

1 Introduction

We are interested in *geospatial language understanding*— the understanding of natural language (NL) descriptions of spatial locations, orientation, movement and paths that are grounded in the real world. Such algorithms would enable a number of applications, including automated geotagging of text and speech, robots that can follow human route instructions, and NL-description based localization.

To aide development of training and testing corpora for this area, we have built The gEoSpatial Language Annotator (TESLA)—a tool which supports the visualization and hand-annotation of both text and speech-based geospatial language corpora. TESLA can be used to create a gold-standard for training and testing geospatial language understanding algorithms by allowing the user to annotate geospatial references with object (e.g., streets, businesses, and parks) and latitude and longitude (lat/lon) coordinates. An integrated search capability to a GIS database with results presented in Google Earth allow the human annotator to easily annotate geospatial references with ground truth.

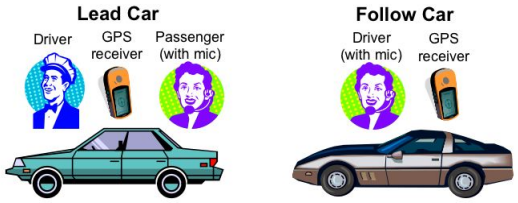


Figure 1: A session in the PURSUIT Corpus

Furthermore, TESLA supports the playback of GPS tracks of multiple objects for corpora associated with synchronized speaker or object movement, allowing the annotator to take this positional context into account. TESLA is currently being used to annotate a corpus of first-person, spoken path descriptions of car routes.

In this paper, we first briefly describe the corpus that we are annotating, which provides a grounded example of using TESLA. We then discuss the TESLA annotation tool and its use in annotating that corpus. Finally, we describe related work and our plans for future work.

2 The PURSUIT Corpus

The PURSUIT Corpus (Blaylock and Allen, 2008) is a collection of speech data in which subjects describe their path in real time (i.e., while they are traveling it) and a GPS receiver simultaneously records the actual paths taken. (These GPS tracks of the actual path can aide the annotator in determining what geospatial entities and events were meant by the speaker's description.)

Figure 1 shows an example of the experimental setup for the corpus collection. Each session consisted of a lead car and a follow car. The driver of the

Proceedings of NAACL HLT 2009: Short Papers, pages 45–48,
Boulder, Colorado, June 2009.

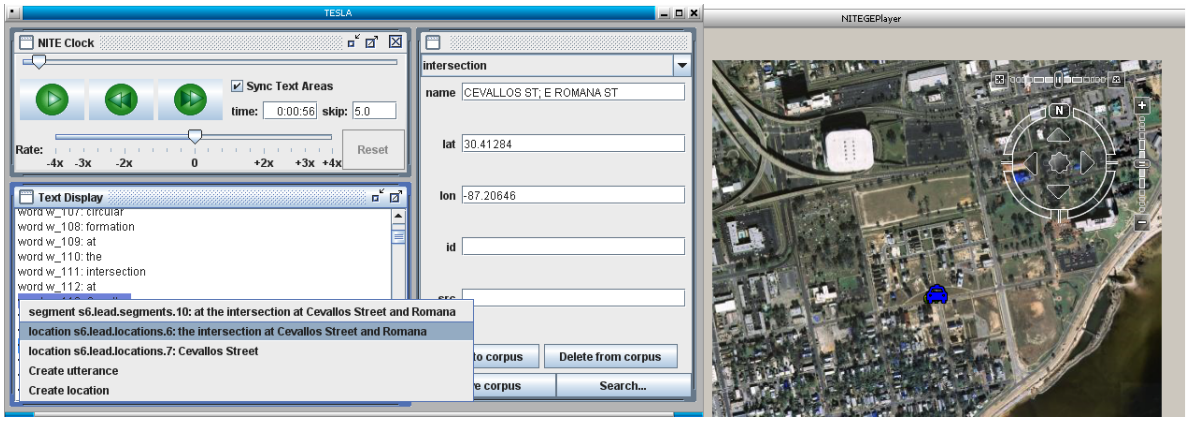


Figure 2: The TESLA annotation and visualization windows

lead car was instructed to drive wherever he wanted for an approximate amount of time (around 15 minutes). The driver of the follow car was instructed to follow the lead car. One person in the lead car (usually a passenger) and one person in the follow car (usually the driver) were given close-speaking headset microphones and instructed to describe, during the ride, where the lead car was going, as if they were speaking to someone in a remote location who was trying to follow the car on a map. The speakers were also instructed to try to be verbose, and that they did not need to restrict themselves to street names—they could use businesses, landmarks, or whatever was natural. Both speakers' speech was recorded during the session. In addition, a GPS receiver was placed in each car and the GPS track was recorded at a high sampling rate. The corpus consists of 13 audio recordings[1] of seven paths along with the corresponding GPS tracks. The average session length was 19 minutes.

3 TESLA

TESLA is an extensible tool for geospatial language annotation and visualization. It is built on the NXT Toolkit (Carletta et al., 2003) and data model (Carletta et al., 2005) and uses Google Earth for visualization. It supports geospatial entity search using the TerraFly GIS database (Rishe et al., 2005). Currently, TESLA supports annotation of geospatial location referring expressions, but is designed to be easily extended to other annotation tasks for geospatial language corpora. (Our plans for extensions are described in Section 6.)

Figure 2 shows a screenshot of the main view in the TESLA annotator, showing a session of the PURSUIT Corpus. In the top-left corner is a widget with playback controls for the session. This provides synchronized playback of the speech and GPS tracks. When the session is playing, audio from a single speaker (lead or follow) is played back, and the blue car icon in the Google Earth window on the right moves in synchronized fashion. Although this Google Earth playback is somewhat analogous to a video of the movement, Google Earth remains usable and the user can move the display or zoom in and out as desired. If location annotations have previously been made, these pop up at the given lat/lon as they are mentioned in the audio, allowing the annotator to verify that the location has been correctly annotated. In the center, on the left-hand side is a display of the audio transcription, which also moves in sync with the audio and Google Earth visualization. The user creates an annotation by highlighting a group of words, and choosing the appropriate type of annotation. The currently selected annotation appears to the right where the corresponding geospatial entity information (e.g., name, address, lat/lon) can be entered by hand, or by searching for the entity in a GIS database.

3.1 GIS Search and Visualization

In addition to allowing information on annotated geospatial entities to be entered by hand, TESLA also supports search with a GIS database. Cur-

[1] In one session, there was no speaker in the lead car.

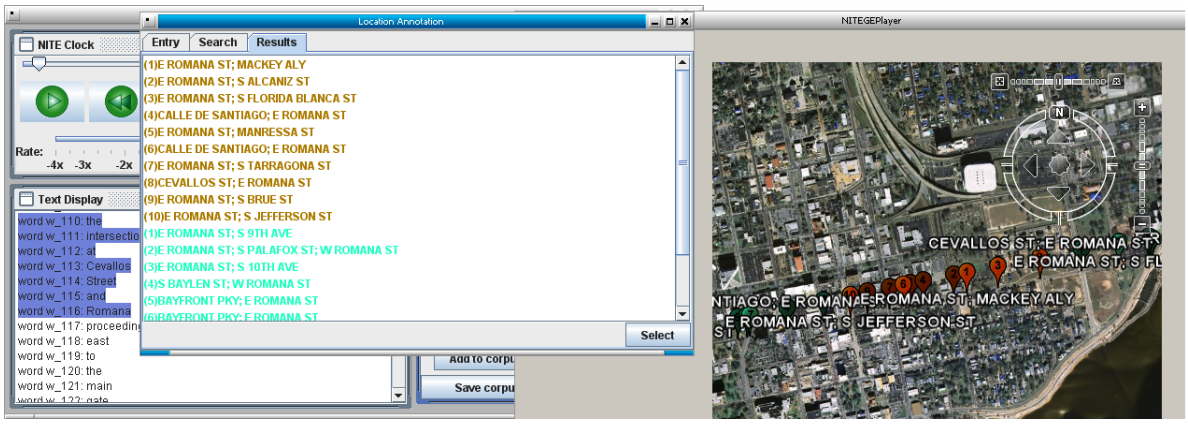


Figure 3: Search results display in TESLA

rently, TESLA supports search queries to the TerraFly database (Rishe et al., 2005), although other databases could be easily added. TerraFly contains a large aggregation of GIS data from major distributors including NavTeq and Tiger streets and roads, 12 million U.S. Businesses through Yellow Pages, and other various freely available geospatial data. It supports keyword searches on database fields as well as radius-bounded searches from a given point. TESLA, by default, uses the position of the GPS track of the car at the time of the utterance as the center for search queries, although any point can be chosen.

Search results are shown to the user in Google Earth as illustrated in Figure 3. This figure shows the result of searching for intersections with the keyword "Romana". The annotator can then select one of the search results, which will automatically populate the geospatial entity information for that annotation. Such visualization is important in geospatial language annotation, as it allows the annotator to verify that the *correct* entity is chosen.

4 Annotation of the PURSUIT Corpus

To illustrate the use of TESLA, we briefly describe our current annotation efforts on the PURSUIT Corpus. We are currently involved in annotating referring expressions to locations in the corpus, although later work will involve annotating movement and orientation descriptions as well.

Location references can occur in a number of syntactic forms, including proper nouns (*Waffle House*), definite (*the street*) and indefinite (*a park*) references, and often, complex noun phrases (*one of the historic churches of Pensacola*). Regardless of its syntactic form, we annotate all references to locations in the corpus that correspond to types found in our GIS database. References to such things as fields, parking lots, and fire hydrants are not annotated, as our database does not contain these types of entities. (Although, with access to certain local government resources or advanced computer vision systems, these references could be resolved as well.) In PURSUIT, we markup the entire noun phrase (as opposed to e.g., the head word) and annotate that grouping.

Rather than annotate a location reference with just latitude and longitude coordinates, we annotate it with the geospatial entity being referred to, such as a street or a business. The reasons for this are twofold: first, lat/lon coordinates are real numbers, and it would be difficult to guarantee that each reference to the same entity was marked with the same coordinates (e.g., to identify coreference). Secondly, targeting the entity allows us to include more information about that entity (as detailed below).

In the corpus, we have found four types of entities that are references, which are also in our database: streets, intersections, addresses (e.g., 127 Main Street), and other points (a catch-all category containing other point-like entities such as businesses, parks, bridges, etc.)

An annotation example is shown in Figure 4, in which the utterance contains references to two

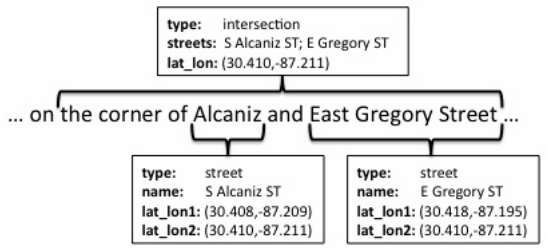


Figure 4: Sample annotations of referring expressions to geospatial locations

streets and an intersection. Here the intersection referring expression spans two referring expressions to streets, and each is annotated with a canonical name as well as lat/lon coordinates. Note also that our annotation schema allows us to annotate embedded references (here the streets within the intersection).

5 Related Work

The SpatialML module for the Callisto annotator (Mani et al., 2008) was designed for human annotation of geospatial locations with ground truth by looking up targets in a gazetteer. It does not, however, have a geographic visualization components such as Google Earth and does not support GPS track playback.

The TAME annotator (Leidner, 2004) is a similar tool, supporting hand annotation of toponym references by gazetteer lookup. It too does not, as far as we are aware, have a visualization component nor GPS track information, likely because the level of geospatial entities being looked at were at the city/state/country level. The PURSUIT Corpus mostly contains references to geospatial entities at a sub-city level, which may introduce more uncertainty as to the intended referent.

6 Conclusion and Future Work

In this paper, we have presented TESLA—a general human annotation tool for geospatial language. TESLA uses a GIS database, GPS tracks, and Google Earth to allow a user to annotate references to geospatial entities. We also discussed how TESLA is being used to annotate a corpus of spoken path descriptions.

Though currently we are only annotating PURSUIT with location references, future plans include extending TESLA to support the annotation of movement, orientation, and path descriptions. We also plan to use this corpus as test and training data for algorithms to automatically annotate such information.

Finally, the path descriptions in the PURSUIT Corpus were all done from a first-person, ground-level perspective. As TESLA allows us to replay the actual routes from GPS tracks within Google Earth, we believe we could use this tool to gather more spoken descriptions of the paths from an aerial perspective from different subjects. This would give us several more versions of descriptions of the same path and allow the comparison of descriptions from the two different perspectives.

References

Nate Blaylock and James Allen. 2008. Real-time path descriptions grounded with gps tracks: a preliminary report. In *LREC Workshop on Methodologies and Resources for Processing Spatial Language*, pages 25–27, Marrakech, Morocco, May 31.

Jean Carletta, Stefan Evert, Ulrich Heid, Jonathan Kilgour, Judy Robertson, and Holger Voormann. 2003. The NITE XML toolkit: flexible annotation for multimodal language data. *Behavior Research Methods, Instruments, and Computers*, 35(3):353–363.

Jean Carletta, Stefan Evert, Ulrich Heid, and Jonathan Kilgour. 2005. The NITE XML toolkit: data model and query language. *Language Resources and Evaluation Journal*, 39(4):313–334.

Jochen L. Leidner. 2004. Towards a reference corpus for automatic toponym resolution evaluation. In *Workshop on Geographic Information Retrieval*, Sheffield, UK.

Inderjeet Mani, Janet Hitzeman, Justin Richer, Dave Harris, Rob Quimby, and Ben Wellner. 2008. SpatialML: Annotation scheme, corpora, and tools. In *6th International Conference on Language Resources and Evaluation (LREC 2008)*, Marrakech, Morocco, May.

N. Rishe, M. Gutierrez, A. Selivonenko, and S. Graham. 2005. TerraFly: A tool for visualizing and dispensing geospatial data. *Imaging Notes*, 20(2):22–23.

Modeling Dialogue Structure with Adjacency Pair Analysis and Hidden Markov Models

Kristy Elizabeth Boyer[*1] **Robert Phillips**[1,2] **Eun Young Ha**[1] **Michael D. Wallis**[1,2] **Mladen A. Vouk**[1] **James C. Lester**[1]

[1]Department of Computer Science
North Carolina State University
Raleigh, NC, USA

[2]Applied Research Associates
Raleigh, NC, USA

[*]keboyer@ncsu.edu

Abstract

Automatically detecting dialogue structure within corpora of human-human dialogue is the subject of increasing attention. In the domain of tutorial dialogue, automatic discovery of dialogue structure is of particular interest because these structures inherently represent tutorial *strategies* or *modes*, the study of which is key to the design of intelligent tutoring systems that communicate with learners through natural language. We propose a methodology in which a corpus of human-human tutorial dialogue is first manually annotated with dialogue acts. Dependent adjacency pairs of these acts are then identified through χ^2 analysis, and hidden Markov modeling is applied to the observed sequences to induce a descriptive model of the dialogue structure.

1 Introduction

Automatically learning dialogue structure from corpora is an active area of research driven by a recognition of the value offered by data-driven approaches (*e.g.,* Bangalore *et al.*, 2006). Dialogue structure information is of particular importance when the interaction is centered around a learning task, such as in natural language tutoring, because techniques that support empirical identification of dialogue strategies can inform not only the design of intelligent tutoring systems (Forbes-Riley *et al.*, 2007), but also contribute to our understanding of the cognitive and affective processes involved in learning through tutoring (VanLehn *et al.*, 2007).

Although traditional top-down approaches (*e.g.,* Cade *et al.*, 2008) and some empirical work on analyzing the structure of tutorial dialogue (Forbes-Riley *et al.*, 2007) have yielded significant results, the field is limited by the lack of an automatic, data-driven approach to identifying dialogue structure. An empirical approach to identifying tutorial dialogue strategies, or *modes*, could address this limitation by providing a mechanism for describing in succinct probabilistic terms the tutorial strategies that actually occur in a corpus.

Just as early work on dialogue act interpretation utilized hidden Markov models (HMMs) to capture linguistic structure (Stolcke *et al.*, 2000), we propose a system that uses HMMs to capture the structure of tutorial dialogue implicit within sequences of already-tagged dialogue acts. This approach operates on the premise that at any given point in the tutorial dialogue, the collaborative interaction is in a dialogue *mode* that characterizes the nature of the exchanges between tutor and student. In our model, a dialogue mode is defined by a probability distribution over the observed symbols (*e.g.*, dialogue acts and adjacency pairs).

Our previous work has noted some limitations of first-order HMMs as applied to sequences of individual dialogue acts (Boyer *et al.*, in press). Chief among these is that HMMs allow arbitrarily frequent transitions between hidden states, which does not conform well to human intuition about how tutoring strategies are applied. Training an HMM on a sequence of adjacency pairs rather than individual dialogue acts is one way to generate a

Proceedings of NAACL HLT 2009: Short Papers, pages 49–52,
Boulder, Colorado, June 2009.

more descriptive model without increasing model complexity more than is required to accommodate the expanded set of observation symbols. To this end, we apply the approach of Midgley *et al.* (2006) for empirically identifying significant adjacency pairs within dialogue, and proceed by treating adjacency pairs as atomic units for the purposes of training the HMM.

2 Corpus Analysis

This analysis uses a corpus of human-human tutorial dialogue collected in the domain of introductory computer science. Forty-three learners interacted remotely with a tutor through a keyboard-to-keyboard remote learning environment yielding 4,864 dialogue moves.

The tutoring corpus was manually tagged with dialogue acts designed to capture the salient characteristics of the tutoring process (Table 1).

Tag	Act	Example
Q	Question	*Where should I declare i?*
EQ	Evaluation Question	*How does that look?*
S	Statement	*You need a closing brace.*
G	Grounding	*Ok.*
EX	Extra-Domain	*You may use your book.*
PF	Positive Feedback	*Yes, that's right.*
LF	Lukewarm Feedback	*Sort of.*
NF	Negative Feedback	*No, that's not right.*

Table 1. Dialogue Act Tags

The correspondence between utterances and dialogue act tags is one-to-one. Compound utterances (*i.e.*, a single utterance comprising more than one dialogue act) were split by the primary annotator prior to the inter-rater reliability study.[1]

The importance of adjacency pairs is well-established in natural language dialogue (*e.g.*, Schlegoff & Sacks, 1973), and adjacency pair analysis has illuminated important phenomena in tutoring as well (Forbes-Riley *et al.*, 2007). For the current corpus, bigram analysis of dialogue acts yielded a set of commonly-occurring pairs. However, as noted in (Midgley *et al.,* 2006), in order to establish that two dialogue acts are truly related as an adjacency pair, it is important to determine whether the presence of the first member of the pair is associated with a significantly higher probability of the second member occurring. For this analysis we utilize a χ^2 test for independence of the categorical variables act_i and act_{i+1} for all two-way combinations of dialogue act tags. Only pairs in which speaker(act_i)≠speaker(act_{i+1}) were considered. Other dialogue acts were treated as atomic elements in subsequent analysis, as discussed in Section 3. Table 2 displays a list of the dependent pairs sorted by descending (unadjusted) statistical significance; the subscript indicates tutor (t) or student (s).

act_i	act_{i+1}	$P(act_{i+1}\mid act_i)$	$P(act_{i+1}\mid \neg act_i)$	χ^2 val	*p*-val
EQ_s	PF_t	0.48	0.07	654	<0.0001
G_s	G_t	0.27	0.03	380	<0.0001
EX_s	EX_t	0.34	0.03	378	<0.0001
EQ_t	PF_s	0.18	0.01	322	<0.0001
EQ_t	S_s	0.24	0.03	289	<0.0001
EQ_s	LF_t	0.13	0.01	265	<0.0001
Q_t	S_s	0.65	0.04	235	<0.0001
EQ_t	LF_s	0.07	0.00	219	<0.0001
Q_s	S_t	0.82	0.38	210	<0.0001
EQ_s	NF_t	0.08	0.01	207	<0.0001
EX_t	EX_s	0.19	0.02	177	<0.0001
NF_s	G_t	0.29	0.03	172	<0.0001
EQ_t	NF_s	0.11	0.01	133	<0.0001
S_s	G_t	0.16	0.03	95	<0.0001
S_s	PF_t	0.30	0.10	90	<0.0001
S_t	G_s	0.07	0.04	36	<0.0001
PF_s	G_t	0.14	0.04	34	<0.0001
LF_s	G_t	0.22	0.04	30	<0.0001
S_t	EQ_s	0.11	0.07	29	<0.0001
G_t	EX_s	0.07	0.03	14	0.002
S_t	Q_s	0.07	0.05	14	0.0002
G_t	G_s	0.10	0.05	9	0.0027
EQ_t	EQ_s	0.13	0.08	8	0.0042

Table 2. Dependent Adjacency Pairs

3 HMM on Adjacency Pair Sequences

The keyboard-to-keyboard tutorial interaction resulted in a sequence of utterances that were annotated with dialogue acts. We have hypothesized that a higher-level dialogue structure, namely the tutorial dialogue *mode*, overlays the observed dialogue acts. To build an HMM model of this struc-

[1] Details of the study procedure used to collect the corpus, as well as Kappa statistics for inter-rater reliability, are reported in (Boyer *et al.*, 2008).

ture we treat dialogue mode as a hidden variable and train a hidden Markov model to induce the dialogue modes and their associated dialogue act emission probability distributions.

An adjacency pair joining algorithm (Figure 1) was applied to each sequence of dialogue acts. This algorithm joins pairs of dialogue acts into atomic units according to a priority determined by the strength of the adjacency pair dependency.

Sort adjacency pair list L by descending statistical significance
For each adjacency pair (act_1, act_2) in L
    For each dialogue act sequence (a_1, a_2, …, a_n) in the corpus
        Replace all pairs (a_i=act_1, a_{i+1}=act_2) with a new single act (act_1act_2)

Figure 1. Adjacency Pair Joining Algorithm

Figure 2 illustrates the application of the adjacency pair joining algorithm on a sequence of dialogue acts. Any dialogue acts that were not grouped into adjacency pairs at the completion of the algorithm are treated as atomic units in the HMM analysis.

Original Dialogue Act Sequence:

Q_s - S_t - LF_t - S_t - S_t - G_s - EQ_s - LF_t - S_t - S_t - Q_s - S_t

After Adjacency Pair Joining Algorithm:

Q_sS_t - LF_t - S_t - **S_tG_s** - **EQ_sLF_t** - S_t - S_t - **Q_sS_t**

Figure 2. DA Sequence Before/After Joining

The final set of observed symbols consists of 39 tags: 23 adjacency pairs (Table 2) plus all individual dialogue acts augmented with a tag for the speaker (Table 1).

It was desirable to learn *n*, the best number of hidden states, during modeling rather than specifying this value *a priori.* To this end, we trained and ten-fold cross-validated seven models (each featuring randomly-initialized parameters) for each number of hidden states *n* from 2 to 15, inclusive.[2] The average log-likelihood was computed across all seven models for each *n*, and this average log-likelihood l_n was used to compute the Akaike Information Criterion, a maximum-penalized likelihood estimator that penalizes more complex models (Scott, 2002). The best fit was obtained with *n*=4 (Figure 3). The transition probability distribution among hidden states is depicted in Figure 4, with the size of the nodes indicating relative frequency of each hidden state; specifically, State 0 accounts for 63% of the corpus, States 1 and 3 account for approximately 15% each, and State 2 accounts for 7%.

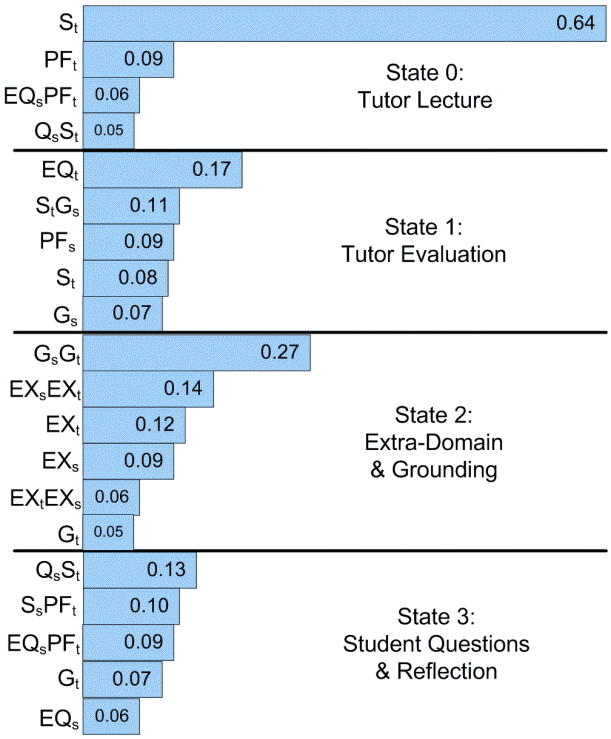


Figure 3. Dialogue Act Emission Probability Distribution by Dialogue Mode[3]

4 Discussion and Future Work

This exploratory application of hidden Markov models involves training an HMM on a mixed input sequence consisting of both individual dialogue acts and adjacency pairs. The best-fit HMM consists of four hidden states whose emission symbol probability distributions lend themselves to interpretation as tutorial dialogue modes. For example, State 0 consists primarily of tutor statements and positive feedback, two of the most common dialogue acts in our corpus. The transition probabili-

[2] *n*=15 was chosen as an initial maximum number of states because it comfortably exceeded our hypothesized range of 3 to 7 (informed by the tutoring literature). The Akaike Information Criterion measure steadily worsened above *n* = 5, confirming no need to train models with *n* > 15.

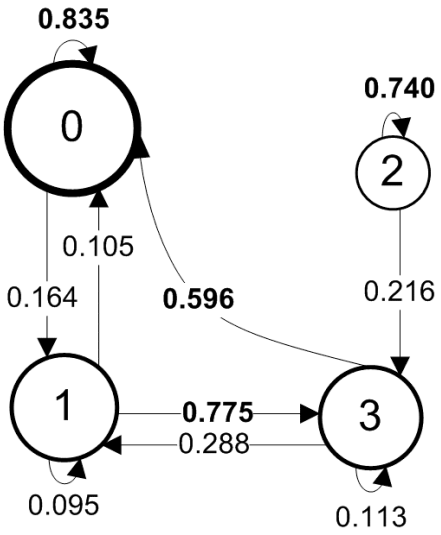


Figure 4. Transition Probability Distribution[4]

ties also reveal that State 0 is highly stable; a self-transition is most likely with probability 0.835. State 3 is an interactive state featuring student reflection in the form of questions, statements, and requests for feedback. The transition probabilities show that nearly 60% of the time the dialogue transitions from State 3 to State 0; this may indicate that after establishing what the student does or does not know in State 3, the tutoring switches to a less collaborative "teaching" mode represented by State 0.

Future evaluation of the HMM presented here will include comparison with other types of graphical models. Another important step is to correlate the dialogue profile of each tutoring session, as revealed by the HMM, to learning and affective outcomes of the tutoring session. This type of inquiry can lead directly to design recommendations for tutorial dialogue systems that aim to maximize particular learner outcomes. In addition, leveraging knowledge of the task state as well as surface-level utterance content below the dialogue act level are promising directions for refining the descriptive and predictive power of these models.

Acknowledgements

This research was supported by the National Science Foundation under Grants REC-0632450, IIS-0812291, CNS-0540523, and GRFP. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

Boyer, K.E., Phillips, R., Wallis, M., Vouk, M., & Lester, J. (2008). Balancing cognitive and motivational scaffolding in tutorial dialogue. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems,* Montreal, Canada, 239-249.

Boyer, K.E., Ha, E.Y., Wallis, M., Phillips, R., Vouk, M. & Lester, J. (in press). Discovering tutorial dialogue strategies with hidden Markov models. To appear in *Proceedings of the 14th International Conference on Artificial Intelligence in Education,* Brighton, U.K.

Bangalore, S., DiFabbrizio, G., Stent, A. (2006). Learning the structure of task-driven human-human dialogs. *Proceedings of ACL*, Sydney, Australia, 201-208.

Cade, W., Copeland, J., Person, N., & D'Mello, S. (2008). Dialog modes in expert tutoring. *Proceedings of the 9th International Conference on Intelligent Tutoring Systems,* Montreal, Canada, 470-479.

Forbes-Riley, K., Rotaru, M., Litman, D. J., & Tetreault, J. (2007). Exploring affect-context dependencies for adaptive system development. *Proceedings of NAACL HLT, Companion Volume,* 41-44.

Midgley, T. D., Harrison, S., & MacNish, C. (2007). Empirical verification of adjacency pairs using dialogue segmentation. *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue,* 104-108.

Schlegoff, E.A., Sacks, H. (1973). Opening up closings. *Semiotica*, *8*(4), 289-327.

Scott, S. L. (2002). Bayesian methods for hidden Markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association, 97*(457), 337-351.

Stolcke, A., Coccaro, N., Bates, R., Taylor, P., Van Ess-Dykema, C., Ries, K., Shirberg, E., Jurafsky, D., Martin, R., Meteer, M. (2000). Dialog act modeling for automatic tagging and recognition of conversational speech. *Computational Linguistics 26*(3), 339-373.

VanLehn, K., Graesser, A., Jackson, G.T., Jordan, P., Olney, A., Rose, C.P. (2007). When are tutorial dialogues more effective than reading? *Cognitive Science, 31*(1), 3-62.

Towards Natural Language Understanding of Partial Speech Recognition Results in Dialogue Systems

Kenji Sagae and **Gwen Christian** and **David DeVault** and **David R. Traum**
Institute for Creative Technologies, University of Southern California
13274 Fiji Way, Marina del Rey, CA 90292
{sagae,gchristian,devault,traum}@ict.usc.edu

Abstract

We investigate natural language understanding of partial speech recognition results to equip a dialogue system with incremental language processing capabilities for more realistic human-computer conversations. We show that relatively high accuracy can be achieved in understanding of spontaneous utterances before utterances are completed.

1 Introduction

Most spoken dialogue systems wait until the user stops speaking before trying to understand and react to what the user is saying. In particular, in a typical dialogue system pipeline, it is only once the user's spoken utterance is complete that the results of automatic speech recognition (ASR) are sent on to natural language understanding (NLU) and dialogue management, which then triggers generation and synthesis of the next system prompt. While this style of interaction is adequate for some applications, it enforces a rigid pacing that can be unnatural and inefficient for mixed-initiative dialogue. To achieve more flexible turn-taking with human users, for whom turn-taking and feedback at the sub-utterance level is natural and common, the system needs to engage in incremental processing, in which interpretation components are activated, and in some cases decisions are made, before the user utterance is complete.

There is a growing body of work on incremental processing in dialogue systems. Some of this work has demonstrated overall improvements in system responsiveness and user satisfaction; e.g. (Aist et al., 2007; Skantze and Schlangen, 2009). Several research groups, inspired by psycholinguistic models of human processing, have also been exploring technical frameworks that allow diverse contextual information to be brought to bear during incremental processing; e.g. (Kruijff et al., 2007; Aist et al., 2007).

While this work often assumes or suggests it is possible for systems to understand partial user utterances, this premise has generally not been given detailed quantitative study. The contribution of this paper is to demonstrate and explore quantitatively the extent to which one specific dialogue system can anticipate what an utterance means, on the basis of partial ASR results, before the utterance is complete.

2 NLU for spontaneous spoken utterances in a dialogue system

For this initial effort, we chose to look at incremental processing of natural language understanding in the SASO-EN system (Traum et al., 2008), a complex spoken dialog system for which we have a corpus of user data that includes recorded speech files that have been transcribed and annotated with a semantic representation. The domain of this system is a negotiation scenario involving the location of a medical clinic in a foreign country. The system is intended as a negotiation training tool, where users learn about negotiation tactics in the context of the culture and social norms of a particular community.

2.1 The natural language understanding task

The NLU module must take the output of ASR as input, and produce domain-specific semantic frames as output. These frames are intended to capture much of the meaning of the utterance, although a

Proceedings of NAACL HLT 2009: Short Papers, pages 53–56,
Boulder, Colorado, June 2009.

dialogue manager further enriches the frame representations with pragmatic information (Traum, 2003). NLU output frames are attribute-value matrices, where the attributes and values are linked to a domain-specific ontology and task model.

Complicating the NLU task of is the relatively high word error rate (0.54) in ASR of user speech input, given conversational speech in a complex domain and an untrained broad user population.

The following example, where the user attempts to address complaints about lack of power in the proposed location for the clinic, illustrates an utterance-frame pair.

- Utterance (speech): *we are prepared to give you guys generators for electricity downtown*
- ASR (NLU input): *we up apparently give you guys generators for a letter city don town*
- Frame (NLU output):

```
<s>.mood declarative
<s>.sem.agent kirk
<s>.sem.event deliver
<s>.sem.modal.possibility can
<s>.sem.speechact.type offer
<s>.sem.theme power-generator
<s>.sem.type event
```

The original NLU component for this system was described in (Leuski and Traum, 2008). For the purposes of this experiment, we have developed a new NLU module and tested on several different data sets as described in the next section. Our approach is to use maximum entropy models (Berger et al., 1996) to learn a suitable mapping from features derived from the words in the ASR output to semantic frames. Given a set of examples of semantic frames with corresponding ASR output, a classifier should learn, for example, that when "generators" appears in the output of ASR, the value *power-generators* is likely to be present in the output frame. The specific features used by the classifier are: each word in the input string (bag-of-words representation of the input), each bigram (consecutive words), each pair of any two words in the input, and the number of words in the input string.

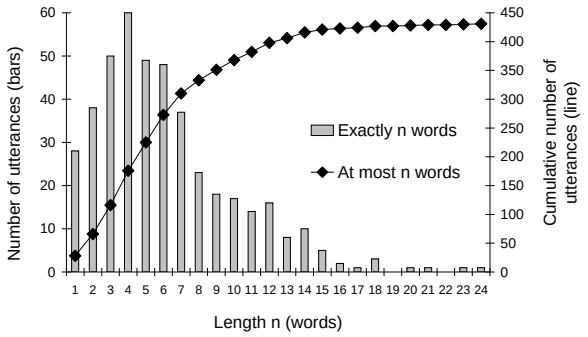


Figure 1: Length of utterances in the development set.

2.2 Data

Our corpus consists of 4,500 user utterances spread across a number of different dialogue sessions. Utterances that were out-of-domain (13.7% of the corpus) were assigned a "garbage" frame, with no semantic content. Approximately 10% of the utterances were set aside for final testing, and another 10% was designated the development corpus for the NLU module. The development and test sets were chosen so that all the utterances in a session were kept in the same set, but sessions were chosen at random for inclusion in the development and test sets.

The training set contains 136 distinct frames, each of which is composed of several attribute-value pairs, called *frame elements*. Figure 1 shows the utterance length distribution in the development set.

2.3 NLU results on complete ASR output

To evaluate NLU results, we look at precision, recall and f-score of frame elements. When the NLU module is trained on complete ASR utterances in the training set, and tested on complete ASR utterances in the development set, f-score of frame elements is 0.76, with precision at 0.78 and recall at 0.74. To gain insight on what the upperbound on the accuracy of the NLU module might be, we also trained the classifier using features extracted from gold-standard manual transcription (instead of ASR output), and tested the accuracy of analyses of gold-standard transcriptions (which would not be available at run-time in the dialogue system). Under these ideal conditions, NLU f-score is 0.87. Training on gold-standard transcriptions and testing on ASR output produces results with a lower f-score, 0.74.

3 NLU on partial ASR results

Roughly half of the utterances in our training data contain six words or more, and the average utterance length is 5.9 words. Since the ASR module is capable of sending partial results to the NLU module even before the user has finished an utterance, in principle the dialogue system can start understanding and even responding to user input as soon as enough words have been uttered to give the system some indication of what the user means, or even what the user will have said once the utterance is completed. To measure the extent to which our NLU module can predict the frame for an input utterance when it sees only a partial ASR result with the first n words, we examine two aspects of NLU with partial ASR results. The first is *correctness* of the NLU output with partial ASR results of varying lengths, if we take the gold-standard manual annotation for the entire utterance as the correct frame for any of the partial ASR results for that utterance. The second is *stability*: how similar the NLU output with partial ASR results of varying lengths is to what the NLU result would have been for the entire utterance.

3.1 Training the NLU module for analysis of partial ASR results

The simplest way to perform NLU of partial ASR results is simply to process the partial utterances using the NLU module trained on complete ASR output. However, better results may be obtained by training separate NLU models for analysis of partial utterances of different lengths. To train these separate NLU models, we first ran the audio of the utterances in the training data through our ASR module, recording all partial results for each utterance. Then, to train a model to analyze partial utterances containing n words, we used only partial utterances in the training set containing n words (unless the entire utterance contained less than n words, in which case we simply used the complete utterance). In some cases, multiple partial ASR results for a single utterance contained the same number of words, and we used the last partial result with the appropriate number of words [1]. We trained separate NLU models for n varying from one to ten.

[1] At run-time, this can be closely approximated by taking the partial utterance immediately preceding the first partial utterance of length $n + 1$.

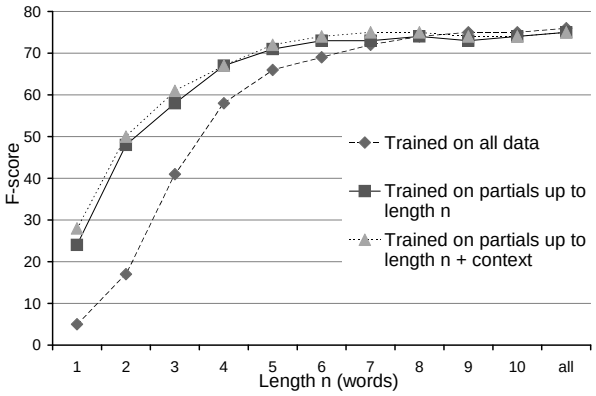


Figure 2: Correctness for three NLU models on partial ASR results up to n words.

3.2 Results

Figure 2 shows the f-score for frames obtained by processing partial ASR results up to length n using three NLU models. The dashed line is our baseline NLU model, trained on complete utterances only (model 1). The solid line shows the results obtained with length-specific NLU models (model 2), and the dotted line shows results for length-specific models that also use features that capture dialogue context (model 3). Models 1 and 2 are described in the previous sections. The additional features used in model 3 are unigram and bigram word features extracted from the most recent system utterance.

As seen in Figure 2, there is a clear benefit to training NLU models specifically tailored for partial ASR results. Training a model on partial utterances with four or five words allows for relatively high f-score of frame elements (0.67 and 0.71, respectively, compared to 0.58 and 0.66 when the same partial ASR results are analyzed using model 1). Considering that half of the utterances are expected to have more than five words (based on the length of the utterances in the training set), allowing the system to start processing user input when four or five-word partial ASR results are available provides interesting opportunities. Targeting partial results with seven words or more is less productive, since the time savings are reduced, and the gain in accuracy is modest.

The context features used in model 3 did not provide substantial benefits in NLU accuracy. It is pos-

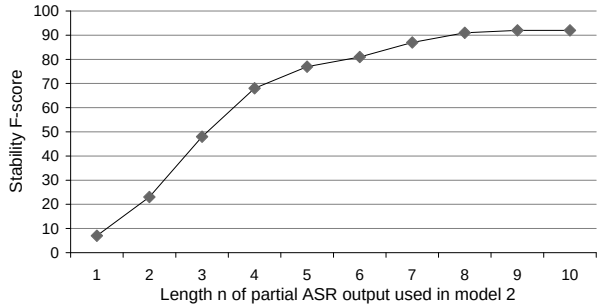


Figure 3: Stability of NLU results for partial ASR results up to length n.

sible that other ways of representing context or dialogue state may be more effective. This is an area we are currently investigating.

Finally, figure 3 shows the *stability* of NLU results produced by model 2 for partial ASR utterances of varying lengths. This is intended to be an indication of how much the frame assigned to a partial utterance differs from the ultimate NLU output for the entire utterance. This ultimate NLU output is the frame assigned by model 1 for the complete utterance. Stability is then measured as the F-score between the output of model 2 for a particular partial utterance, and the output of model 1 for the corresponding complete utterance. A stability F-score of 1.0 would mean that the frame produced for the partial utterance is identical to the frame produced for the entire utterance. Lower values indicate that the frame assigned to a partial utterance is revised significantly when the entire input is available. As expected, the frames produced by model 2 for partial utterances with at least eight words match closely the frames produced by model 1 for the complete utterances. Although the frames for partial utterances of length six are almost as accurate as the frames for the complete utterances (figure 2), figure 3 indicates that these frames are still often revised once the entire input utterance is available.

4 Conclusion

We have presented experiments that show that it is possible to obtain domain-specific semantic representations of spontaneous speech utterances with reasonable accuracy before automatic speech recognition of the utterances is completed. This allows for interesting opportunities in dialogue systems, such as agents that can interrupt the user, or even finish the user's sentence. Having an estimate of the correctness and stability of NLU results obtained with partial utterances allows the dialogue system to estimate how likely its initial interpretation of an user utterance is to be correct, or at least agree with its ultimate interpretation. We are currently working on the extensions to the NLU model that will allow for the use of different types of context features, and investigating interesting ways in which agents can take advantage of early interpretations.

Acknowledgments

The work described here has been sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

References

G. Aist, J. Allen, E. Campana, C. G. Gallo, S. Stoness, M. Swift, and M. K. Tanenhaus. 2007. Incremental dialogue system faster than and preferred to its non-incremental counterpart. In *Proc. of the 29th Annual Conference of the Cognitive Science Society*.

A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra. 1996. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71.

G. J. Kruijff, P. Lison, T. Benjamin, H. Jacobsson, and N. Hawes. 2007. Incremental, multi-level processing for comprehending situated dialogue in human-robot interaction. In *Language and Robots: Proc. from the Symposium (LangRo'2007)*. University of Aveiro, 12.

A. Leuski and D. Traum. 2008. A statistical approach for text processing in virtual humans. In *26th Army Science Conference*.

G. Skantze and D. Schlangen. 2009. Incremental dialogue processing in a micro-domain. In *Proc. of the 12th Conference of the European Chapter of the ACL*.

D. Traum, S. Marsella, J. Gratch, J. Lee, and A. Hartholt. 2008. Multi-party, multi-issue, multi-strategy negotiation for multi-modal virtual agents. In *Proc. of Intelligent Virtual Agents Conference IVA-2008*.

D. Traum. 2003. Semantics and pragmatics of questions and answers for dialogue agents. In *Proc. of the International Workshop on Computational Semantics*, pages 380–394, January.

Spherical Discriminant Analysis in Semi-supervised Speaker Clustering*

Hao Tang
Dept. of ECE
University of Illinois
Urbana, IL 61801, USA
haotang2@ifp.uiuc.edu

Stephen M. Chu
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598, USA
schu@us.ibm.com

Thomas S. Huang
Dept. of ECE
University of Illinois
Urbana, IL 61801, USA
huang@ifp.uiuc.edu

Abstract

Semi-supervised speaker clustering refers to the use of our prior knowledge of speakers in general to assist the unsupervised speaker clustering process. In the form of an independent training set, the prior knowledge helps us learn a speaker-discriminative feature transformation, a universal speaker prior model, and a discriminative speaker subspace, or equivalently a speaker-discriminative distance metric. The directional scattering patterns of Gaussian mixture model mean supervectors motivate us to perform discriminant analysis on the unit hypersphere rather than in the Euclidean space, which leads to a novel dimensionality reduction technique called spherical discriminant analysis (SDA). Our experiment results show that in the SDA subspace, speaker clustering yields superior performance than that in other reduced-dimensional subspaces (e.g., PCA and LDA).

1 Introduction

Speaker clustering is a critical part of speaker diarization (a.k.a. speaker segmentation and clustering) (Barras et al., 2006; Tranter and Reynolds, 2006; Wooters and Huijbregts, 2007; Han et al., 2008). Unlike speaker recognition, where we have the training data of a set of known speakers and thus recognition can be done supervised, speaker clustering is usually performed in a completely unsupervised manner. The output of speaker clustering is the internal labels relative to a dataset rather than real speaker identities. An interesting question is: Can we do semi-supervised speaker clustering? That is, can we make use of any available information that can be helpful to speaker clustering?

Our answer to this question is positive. Here, semi-supervision refers to the use of our prior knowledge of speakers in general to assist the unsupervised speaker clustering process. In the form of an independent training set, the prior knowledge helps us learn a speaker-discriminative feature transformation, a universal speaker prior model, and a discriminative speaker subspace, or equivalently a speaker-discriminative distance metric.

2 Semi-supervised Speaker Clustering

A general pipeline of speaker clustering consists of four essential elements, namely feature extraction, utterance representation, distance metric, and clustering. We incorporate our prior knowledge of speakers into the various stages of this pipeline through an independent training set.

2.1 Feature Extraction

The most popular speech features are spectrum-based acoustic features such as mel-frequency cepstral coefficients (MFCCs) and perceptual linear predictive (PLP) coefficients. In order to account for the dynamics of spectrum changes over time, the basic acoustic features are often supplemented by their first and second derivatives. We pursue a different avenue in which we augment the basic acoustic features of every frame with those of the neighboring frames. Specifically, the acoustic features of the current frame and those of the K_L frames

*This work was funded in part by DARPA contract HR0011-06-2-0001.

Proceedings of NAACL HT 2009: Short Papers, pages 57–60,
Boulder, Colorado, June 2009.

to the left and K_R frames to the right are concatenated to form a high-dimensional feature vector. In the context-expanded feature vector space, we learn a speaker-discriminative feature transformation by linear discriminant analysis (LDA) based on the known speaker labels of the independent training set. The resulting low-dimensional feature subspace is expected to provide optimal speaker separability.

2.2 Utterance Representation

Deviating from the mainstream "bag of acoustic features" representation where the extracted acoustic features are represented by a statistical model such as a Gaussian mixture model (GMM), we adopt the GMM mean supervector representation which has emerged in the speaker recognition area (Campbell et al., 2006). Such representation is obtained by *maximum a posteriori* (MAP) adapting a universal background model (UBM), which has been finely trained with all the data in the training set, to a particular utterance. The component means of the adapted GMM are stacked to form a column vector conventionally called a GMM mean supervector. In this way, we are allowed to represent an utterance as a point in a high-dimensional space where traditional distance metrics and clustering techniques can be naturally applied. The UBM, which can be deemed as a universal speaker prior model inferred from the independent training set, imposes generic speaker constraints to the GMM mean supervector space.

2.3 Distance Metric

In the GMM mean supervector space, a naturally arising distance metric is the Euclidean distance metric. However, it is observed that the supervectors show strong directional scattering patterns. The directions of the data points seem to be more indicative than their magnitudes. This observation motivates us to favor the cosine distance metric over the Euclidean distance metric for speaker clustering.

Although the cosine distance metric can be used in the GMM mean supervector space, it is optimal only if the data points are uniformly spread in all directions in the entire space. In a high-dimensional space, most often the data lies in or near a low-dimensional manifold or subspace. It is advantageous to learn an optimal distance metric from the data directly.

The general cosine distance between two data points $\mathbf{x}$ and $\mathbf{y}$ can be defined and manipulated as follows.

$$\begin{aligned} d(\mathbf{x},\mathbf{y}) &= 1 - \frac{\mathbf{x}^T A \mathbf{y}}{\sqrt{\mathbf{x}^T A \mathbf{x}}\sqrt{\mathbf{y}^T A \mathbf{y}}} \\ &= 1 - \frac{(A^{1/2}\mathbf{x})^T (A^{1/2}\mathbf{y})}{\sqrt{(A^{1/2}\mathbf{x})^T (A^{1/2}\mathbf{x})}\sqrt{(A^{1/2}\mathbf{y})^T (A^{1/2}\mathbf{y})}} \\ &= 1 - \frac{(W^T\mathbf{x})^T (W^T\mathbf{y})}{\sqrt{(W^T\mathbf{x})^T (W^T\mathbf{x})}\sqrt{(W^T\mathbf{y})^T (W^T\mathbf{y})}} \end{aligned} \quad (1)$$

The general cosine distance can be casted as the cosine distance between two transformed data points $W^T\mathbf{x}$ and $W^T\mathbf{y}$ where $W^T = A^{1/2}$. In this sense, learning an optimal distance metric is equivalent to learning an optimal linear subspace of the original high-dimensional space.

3 Spherical Discriminant Analysis

Most existing linear subspace learning techniques (e.g. PCA and LDA) are based on the Euclidean distance metric. In the GMM mean supervector space, we seek to perform discriminant analysis in the cosine distance metric space. We coin the phrase "spherical discriminant analysis" to denote discriminant analysis on the unit hypersphere. We define a projection from a d-dimensional hypersphere to a d'-dimensional hypersphere where $d' < d$

$$\mathbf{y} = \frac{W^T\mathbf{x}}{\|W^T\mathbf{x}\|} \quad (2)$$

We note that such a projection is nonlinear. However, under two mild conditions, this projection can be linearized. One is that the objective function for learning the projection only involves the cosine distance. The other is that only the cosine distance is used in the projected space. In this case, the norm of the projected vector $\mathbf{y}$ has no impact on the objective function and distance computation in the projected space. Thus, the denominator term of Equation 2 can be safely dropped, leading to a linear projection.

3.1 Formulation

The goal of SDA is to seek a linear transformation W such that the average within-class cosine similarity of the projected data set is maximized while the

average between-class cosine similarity of the projected data set is minimized. Assuming that there are c classes, the average within-class cosine similarity can be written in terms of the unknown projection matrix W and the original data points $\mathbf{x}$

$$S_W = \frac{1}{c}\sum_{i=1}^{c} S_i \tag{3}$$

$$\begin{aligned} S_i &= \frac{1}{|D_i||D_i|}\sum_{\mathbf{y}_j,\mathbf{y}_k\in D_i}\frac{\mathbf{y}_j^T\mathbf{y}_k}{\sqrt{\mathbf{y}_j^T\mathbf{y}_j}\sqrt{\mathbf{y}_k^T\mathbf{y}_k}} \\ &= \frac{1}{|D_i||D_i|}\sum_{\mathbf{x}_j,\mathbf{x}_k\in D_i}\frac{\mathbf{x}_j^T WW^T\mathbf{x}_k}{\sqrt{\mathbf{x}_j^T WW^T\mathbf{x}_j}\sqrt{\mathbf{x}_k^T WW^T\mathbf{x}_k}} \end{aligned}$$

where $|D_i|$ denotes the number of data points in the i^{th} class. Similarly, the average between-class cosine similarity can be written in terms of W and $\mathbf{x}$

$$S_B = \frac{1}{c(c-1)}\sum_{m=1}^{c}\sum_{n=1}^{c} S_{mn} \quad (m\neq n) \tag{4}$$

$$\begin{aligned} S_{mn} &= \frac{1}{|D_m||D_n|}\sum_{\substack{\mathbf{y}_j\in D_m\\ \mathbf{y}_k\in D_n}}\frac{\mathbf{y}_j^T\mathbf{y}_k}{\sqrt{\mathbf{y}_j^T\mathbf{y}_j}\sqrt{\mathbf{y}_k^T\mathbf{y}_k}} \\ &= \frac{1}{|D_m||D_n|}\sum_{\substack{\mathbf{x}_j\in D_m\\ \mathbf{x}_k\in D_n}}\frac{\mathbf{x}_j^T WW^T\mathbf{x}_k}{\sqrt{\mathbf{x}_j^T WW^T\mathbf{x}_j}\sqrt{\mathbf{x}_k^T WW^T\mathbf{x}_k}} \end{aligned}$$

where $|D_m|$ and $|D_n|$ denote the number of data points in the m^{th} and n^{th} classes, respectively.

The SDA criterion is to maximize S_W while minimizing S_B

$$W = arg\max_W (S_W - S_B) \tag{5}$$

Our SDA formulation is similar to the work of Ma et al. (2007). However, we solve it efficiently in a general dimensionality reduction framework known as graph embedding (Yan et al., 2007).

3.2 Graph Embedding Solution

In graph embedding, a weighted graph with vertex set X and similarity matrix S is used to characterize certain statistical or geometrical properties of a data set. A vertex in X represents a data point and an entry s_{ij} in S represents the similarity between the data points $\mathbf{x}_i$ and $\mathbf{x}_j$. For a specific dimensionality reduction algorithm, there may exist two graphs. The intrinsic graph $\{X, S^{(i)}\}$ characterizes the data properties that the algorithm aims to preserve and the penalty graph $\{X, S^{(p)}\}$ characterizes the data properties that the algorithm aims to avoid. The goal of graph embedding is to represent each vertex in X as a low dimensional vector that preserves the similarities in S. The objective function is

$$W = arg\min_W \sum_{i\neq j} \|f(\mathbf{x}_i,W) - f(\mathbf{x}_j,W)\|^2 (s_{ij}^{(i)} - s_{ij}^{(p)}) \tag{6}$$

where $f(\mathbf{x}, W)$ is a general projection with parameters W. If we take the projection to be of the form in Equation 2, the objective function becomes

$$W = arg\min_W \sum_{i\neq j} \left\| \frac{W^T\mathbf{x}_i}{\|W^T\mathbf{x}_i\|} - \frac{W^T\mathbf{x}_j}{\|W^T\mathbf{x}_j\|} \right\|^2 (s_{ij}^{(i)} - s_{ij}^{(p)}) \tag{7}$$

It is shown that the solution to the graph embedding problem of Equation 7 may be obtained by a steepest descent algorithm (Fu et al., 2008). If we expand the L_2 norm terms of Equation 7, it is straightforward to show that Equation 7 is equivalent to Equation 5 provided that the graph weights are set to proper values, as follows.

$$\begin{aligned} s_{jk}^{(i)} &\leftarrow \frac{1}{c|D_i||D_i|} \quad \text{if } \mathbf{x}_j, \mathbf{x}_k \in D_i, \quad i = 1, ..., c \\ s_{jk}^{(p)} &\leftarrow \frac{1}{c(c-1)|D_m||D_n|} \quad \text{if } \mathbf{x}_j \in D_m, \mathbf{x}_k \in D_n \\ & \quad m, n = 1, ..., c, m \neq n \end{aligned} \tag{8}$$

That is, by assigning appropriate values to the weights of the intrinsic and penalty graphs, the SDA optimization problem in Equation 5 can be solved within the elegant graph embedding framework.

4 Experiments

Our speaker clustering experiments are based on a test set of 630 speakers and 19024 utterances selected from the GALE database (Chu et al., 2008), which contains about 1900 hours of broadcasting news speech data collected from various TV programs. An independent training set of 498 speakers and 18327 utterances is also selected from the GALE database. In either data set, there are an average of 30-40 utterances per speaker and the average duration of the utterances is about 3-4 seconds. Note that there are no overlapping speakers in the two data

sets – speakers in the test set are not present in the independent training set.

The acoustic features are 13 basic PLP features with cepstrum mean subtraction. In computing the LDA feature transformation using the independent training set, K_L and K_R are both set to 4, and the dimensionality of the low-dimensional feature space is set to 40. The entire independent training set is used to train a UBM via the EM algorithm, and a GMM mean supervector is obtained for every utterance in the test set via MAP adaptation. The trained UBM has 64 mixture components. Thus, the dimension of the GMM mean supervectors is 2560.

We employ the hierarchical agglomerative clustering technique with the "ward" linkage method. Our experiments are carried out as follows. In each experiment, we perform 4 cases, each of which is associated with a specific number of test speakers, i.e., 5, 10, 20, and 50, respectively. In each case, the corresponding number of speakers are drawn randomly from the test set, and all the utterances from the selected speakers are used for clustering. For each case, 100 trials are run, each of which involves a random draw of the test speakers, and the average of the clustering accuracies across the 100 trials is recorded.

First, we perform speaker clustering in the original GMM mean supervector space using the Euclidean distance metric and the cosine distance metric, respectively. The results indicate that the cosine distance metric consistently outperforms the Euclidean distance metric. Next, we perform speaker clustering in the reduced-dimensional subspaces using the eigenvoice (PCA) and fishervoice (LDA) approaches, respectively. The results show that the fishervoice approach significantly outperforms the eigenvoice approach in all cases. Finally, we perform speaker clustering in the SDA subspace. The results demonstrate that in the SDA subspace, speaker clustering yields superior performance than that in other reduced-dimensional subspaces (e.g., PCA and LDA). Table 1 presents these results.

Metric	Subspace	5	10	20	50
Euc	Orig	85.0	82.6	78.1	69.4
	PCA	85.5	82.9	79.3	69.9
	LDA	94.0	90.8	86.6	79.6
Cos	Orig	90.7	86.5	82.2	77.7
	SDA	**98.0**	**94.7**	**90.0**	**85.9**

Table 1: Average speaker clustering accuracies (unit:%).

5 Conclusion

This paper proposes semi-supervised speaker clustering in which we learn a speaker-discriminative feature transformation, a universal speaker prior model, and a speaker-discriminative distance metric through an independent training set. Motivated by the directional scattering patterns of the GMM mean supervectors, we peroform discriminant analysis on the unit hypersphere rather than in the Euclidean space, leading to a novel dimensionality reduction technique "SDA". Our experiment results indicate that in the SDA subspace, speaker clustering yields superior performance than that in other reduced-dimensional subspaces (e.g., PCA and LDA).

References

C. Barras, X. Zhu, S. Meignier, and J. Gauvain. 2006. Multistage speaker diarization of broadcast news. *IEEE Trans. ASLP*, 14(5):1505–1512.

W. Campbell, D. Sturim, D. Reynolds. 2006. Support vector machines using GMM supervectors for speaker verification. *Signal Processing Letters* 13(5):308-311.

S. Chu, H. Kuo, L. Mangu, Y. Liu, Y. Qin, and Q. Shi. 2008. Recent advances in the IBM GALE mandarin transcription system. *Proc. ICASSP*.

Y. Fu, S. Yan and T. Huang. 2008. Correlation Metric for Generalized Feature Extraction. *IEEE Trans. PAMI* 30(12):2229–2235.

K. Han, S. Kim, and S. Narayanan. 2008. Strategies to Improve the Robustness of Agglomerative Hierarchical Clustering under Data Source Variation for Speaker Diarization. *IEEE Trans. SALP* 16(8):1590–1601.

Y. Ma, S. Lao, E. Takikawa, and M. Kawade. 2007. Discriminant Analysis in Correlation Similarity Measure Space. *Proc. ICML* (227):577–584.

S. Tranter and D. Reynolds. 2006. An Overview of Automatic Speaker Diarization Systems. *IEEE Trans. ASLP*, 14(5):1557–1565.

C. Wooters and M. Huijbregts. 2007. The ICSI RT07s Speaker Diarization System. *LNCS*.

S. Yan, D. Xu, B. Zhang, H. Zhang, Q. Yang, S. Lin. 2007. Graph Embedding and Extensions: A General Framework for Dimensionality Reduction. *IEEE Trans. PAMI* 29(1):40–51.

Learning Bayesian Networks for Semantic Frame Composition in a Spoken Dialog System

Marie-Jean Meurs, Fabrice Lefèvre and Renato de Mori
Université d'Avignon et des Pays de Vaucluse
Laboratoire Informatique d'Avignon (EA 931), F-84911 Avignon, France.
{marie-jean.meurs,fabrice.lefevre,renato.demori}@univ-avignon.fr

Abstract

A stochastic approach based on Dynamic Bayesian Networks (DBNs) is introduced for spoken language understanding. DBN-based models allow to infer and then to compose semantic frame-based tree structures from speech transcriptions. Experimental results on the French MEDIA dialog corpus show the appropriateness of the technique which both lead to good tree identification results and can provide the dialog system with n-best lists of scored hypotheses.

1 Introduction

Recent developments in Spoken Dialog Systems (SDSs) have renewed the interest for the extraction of rich and high-level semantics from users' utterances. Shifting every SDS component from hand-crafted to stochastic is foreseen as a good option to improve their overall performance by an increased robustness to speech variabilities. For instance stochastic methods are now efficient alternatives to rule-based techniques for Spoken Language Understanding (SLU) (He and Young, 2005; Lefèvre, 2007).

The SLU module links up the automatic speech recognition (ASR) module and the dialog manager. From the user's utterance analysis, it derives a representation of its semantic content upon which the dialog manager can decide the next best action to perform, taking into account the current dialog context. In this work, the overall objective is to increase the relevancy of the semantic information used by the system. Generally the internal meaning representation is based on flat concept sets obtained by either keyword spotting or conceptual decoding. In some cases a dialog act can be added on top of the concept set. Here we intend to consider an additional semantic composition step which will capture the abstract semantic structures conveyed by the basic concept representation. A frame formalism is applied to specify these nested structures. As such structures do not rely on sequential constraints, pure left-right branching semantic parser (such as (He and Young, 2005)) will not apply in this case.

To derive automatically such frame meaning representations we propose a system based on a two decoding step process using dynamic Bayesian networks (DBNs) (Bilmes and Zweig, 2002): first basic concepts are derived from the user's utterance transcriptions, then inferences are made on sequential semantic frame structures, considering all the available previous annotation levels (words and concepts). The inference process extracts all possible sub-trees (branches) according to lower level information (*generation*) and composes the hypothesized branches into a single utterance-span tree (*composition*). A hand-craft rule-based approach is used to derive the seed annotated training data. So both approaches are not competing and the stochastic approach is justified as only the DBN system is able to provide n-best lists of tree hypotheses with confidence scores to a stochastic dialog manager (such as the very promising POMDP-based approaches).

The paper is organized as follows. The next section presents the semantic frame annotation on the MEDIA corpus. Then Section 3 introduces the DBN-based models for semantic composition and finally Section 4 reports on the experiments.

Proceedings of NAACL HLT 2009: Short Papers, pages 61–64,
Boulder, Colorado, June 2009.

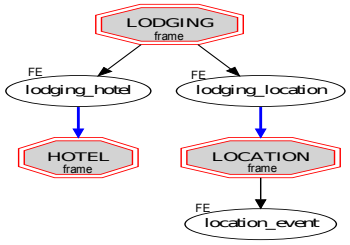


Figure 1: Frames, FEs and relations associated to the sequence *"staying in a hotel near the Festival de Cannes"*

2 Semantic Frames on the MEDIA corpus

MEDIA is a French corpus of negotiation dialogs among users and a tourist information phone server (Bonneau-Maynard et al., 2005). The corpus contains 1,257 dialogs recorded using a *Wizard of Oz* system. The semantic corpus is annotated with *concept-value* pairs corresponding to word segments with the addition of *specifier* tags representing some relations between concepts. The annotation utilizes 83 basic concepts and 19 specifiers.

Amongst the available semantic representations, the semantic frames (Lowe et al., 1997) are probably the most suited to the task, mostly because of their ability to represent negotiation dialogs. Semantic frames are computational models describing common or abstract situations involving roles, the frame elements (FEs). The FrameNet project (Fillmore et al., 2003) provides a large frame database for English. As no such resource exists for French, we elaborated a frame ontology to describe the semantic knowledge of the MEDIA domain. The MEDIA ontology is composed of 21 frames and 86 FEs. All are described by a set of manually defined patterns made of lexical units and conceptual units (frame and FE evoking words and concepts). Figure 1 gives the annotation of word sequence *"staying in a hotel near the Festival de Cannes"*. The training data are automatically annotated by a rule-based process. Pattern matching triggers the instantiation of frames and FEs which are composed using a set of logical rules. Composition may involve creation, modification or deletion of frame and FE instances. About 70 rules are currently used. This process is task-oriented and is progressively enriched with new rules to improve its accuracy. A reference frame annotation for the training corpus is established in this way and used for learning the parameters of the stochastic models introduced in the next section.

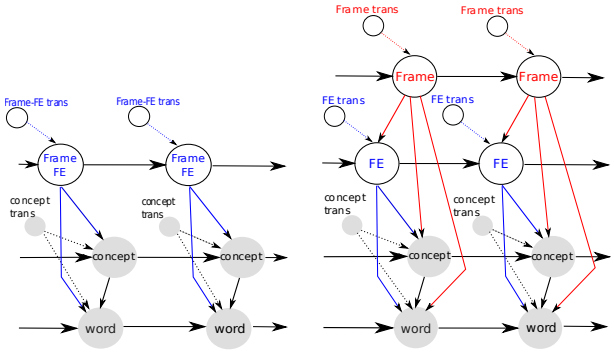


Figure 2: Frames, FEs as one or 2 unobserved variables

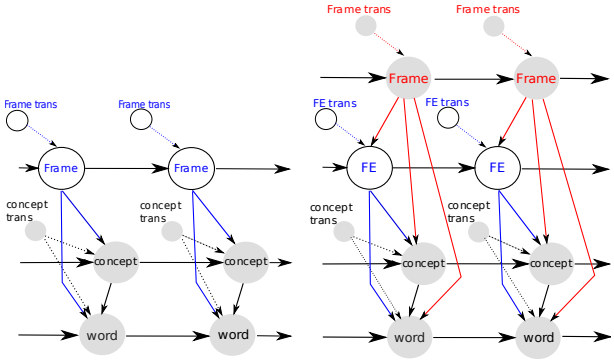


Figure 3: 2-level decoding of frames and FEs

3 DBN-based Frame Models

The generative DBN models used in the system are depicted on two time slices (two words) in figures 2 and 3. In practice, a regular pattern is repeated sufficiently to fit the entire word sequence. Shaded nodes are observed variables whereas empty nodes are hidden. Plain lines represent conditional dependencies between variables and dashed lines indicate switching parents (variables modifying the conditional relationship between others). An example of a switching parent is given by the `trans` nodes which influence the frame and FE nodes: when `trans` node is null the frame or FE stays the same from slice to slice, when `trans` is 1 a new frame or FE value is predicted based on the values of its parent nodes in the word sequence using frame (or FE) n-grams.

In the left DBN model of Figure 2 frames and FEs are merged in a single compound variable. They are factorized in the right model using two variables jointly decoded. Figure 3 shows the 2-level model where frames are first decoded then used as observed values in the FE decoding step. Merging frames and FEs into a variable reduces the decoding complexity but leads to deterministic links between frames

and FEs. With their factorization, on the contrary, it is possible to deal with the ambiguities in the frame and FE links. During the decoding step, every combination is tested, even not encountered in the training data, by means of a back-off technique. Due to the increase in model complexity, a sub-optimal beam search is applied for decoding. In this way, the 2-level approach reduces the complexity of the factored approach while preserving model generalization.

Because all variables are observed at training time, the edge's conditional probability tables are directly derived from observation counts. To improve their estimates, factored language models (FLMs) are used along with generalized parallel backoff (Bilmes and Kirchhoff, 2003). Several FLM implementations of the joint distributions are used in the DBN models, corresponding to the arrows in Figures 2 and 3. In the FLMs given below, n is the history length ($n = 1$ for bigrams), the uppercase and lowercase letters FFE, F, FE, C and W respectively stand for frame/FE (one variable), frame, FE, concept and word variables:

• Frame/FE compound variable:

$P(FFE) \simeq \prod_{k=0}^{n} P(ffe_k|ffe_{k-1})$;

$P(C|FFE) \simeq \prod_{k=0}^{n} P(c_k|c_{k-1}, ffe_k)$;

$P(W|C, FFE) \simeq \prod_{k=0}^{n} P(w_k|w_{k-1}, c_k, ffe_k)$.

• Frame and FE variables, joint decoding:

$P(F) \simeq \prod_{k=0}^{n} P(f_k|f_{k-1})$;

$P(FE|F) \simeq \prod_{k=0}^{n} P(fe_k|fe_{k-1}, f_k)$;

$P(C|FE, F) \simeq \prod_{k=0}^{n} P(c_k|c_{k-1}, fe_k, f_k)$;

$P(W|C, FE, F) \simeq \prod_{k=0}^{n} P(w_k|w_{k-1}, c_k, fe_k, f_k)$.

• Frame and FE variables, 2-level decoding:

– *First stage:* same as frame/FE compound variables but only decoding frames

– *Second stage:* same as joint decodind but frames are observed

$P(\hat{F}) \simeq \prod_{k=0}^{n} P(\hat{f}_k|\hat{f}_{k-1})$;

$P(FE|\hat{F}) \simeq \prod_{k=0}^{n} P(fe_k|fe_{k-1}, \hat{f}_k)$;

$P(C|\hat{F}, FE) \simeq \prod_{k=0}^{n} P(c_k|c_{k-1}, \hat{f}_k, fe_k)$;

$P(W|C, \hat{F}, FE) \simeq \prod_{k=0}^{n} P(w_k|w_{k-1}, c_k, \hat{f}_k, fe_k)$.

Variables with hat have observed values.

Due to the frame hierarchical representation, some overlapping situations can occurred when determining the frame and FE associated to a concept. To address this difficulty, a tree-projection algorithm is performed on the utterance tree-structured frame annotation and allows to derive sub-branches associated to a concept (possibly more than one). Starting from a leaf of the tree, a compound frame/FE class is obtained by aggregating the father vertices (either frames or FEs) as long as they are associated to the same concept (or none). The edges are defined both by the frame→FE attachments and the FE→frame sub-frame relations.

Thereafter, either the branches are considered directly as compound classes or the frame and FE interleaved components are separated to produce two class sets. These compound classes are considered in the decoding process then projected back afterwards to recover the two types of frame↔FE connections. However, some links are lost because decoding is sequential. A set of manually defined rules is used to retrieve the missing connections from the set of hypothesized branches. Theses rules are similar to those used in the semi-automatic annotation of the training data but differ mostly because the available information is different. For instance, the frames cannot anymore be associated to a particular word inside a concept but rather to the whole segment. The training corpus provides the set of frame and FE class sequences on which the DBN parameters are estimated.

4 Experiments and Results

The DBN-based composition systems were evaluated on a test set of 225 speakers' turns manually annotated in terms of frames and FEs. The rule-based system was used to perform a frame annotation of the MEDIA data. On the test set, an average F-measure of 0.95 for frame identification confirms the good reliability of the process. The DBN model parameters were trained on the training data using jointly the manual transcriptions, the manual concept annotations and the rule-based frame annotations.

Experiments were carried out on the test set under three conditions varying the input noise level:

• REF (reference): speaker turns manually transcribed and annotated;

• SLU: concepts decoded from manual transcriptions using a DBN-based SLU model comparable to (Lefèvre, 2007) (10.6% concept error rate);

• ASR+SLU: 1-best hypotheses of transcriptions

Inputs		REF			SLU			ASR + SLU		
DBN models		Frames	*FE*	**Links**	Frames	*FE*	**Links**	Frames	*FE*	**Links**
frame/FEs	$\bar{p}/\bar{r}$	0.91/0.93	*0.91*/0.86	**0.93/0.98**	0.87/0.82	*0.91*/0.83	**0.93/0.98**	0.86/0.80	*0.90*/0.86	**0.92/0.98**
(compound)	$\bar{F}$**-m**	**0.89**	***0.86***	**0.92**	**0.81**	***0.82***	**0.92**	**0.78**	***0.84***	**0.92**
frames and FEs	$\bar{p}/\bar{r}$	0.92/0.92	*0.92*/0.85	**0.94/0.98**	0.88/0.81	*0.92*/0.83	**0.93/0.97**	0.87/0.79	*0.90*/0.86	**0.94/0.97**
(2 variables)	$\bar{F}$**-m**	**0.90**	***0.86***	**0.94**	**0.80**	***0.83***	**0.91**	**0.78**	***0.84***	**0.93**
frames then FEs	$\bar{p}/\bar{r}$	0.92/0.94	*0.91*/0.82	**0.92/0.98**	0.88/0.86	*0.91*/0.80	**0.92/0.97**	0.87/0.81	*0.89*/0.82	**0.93/0.98**
(2-level)	$\bar{F}$**-m**	**0.91**	***0.83***	**0.93**	**0.83**	***0.80***	**0.90**	**0.79**	***0.80***	**0.92**

Table 1: Precision ($\bar{p}$), Recall ($\bar{r}$) and F-measure ($\bar{F}$-m) on the MEDIA test set for the DBN-based frame composition systems.

generated by an ASR system and concepts decoded using them (14.8% word error rate, 24.3% concept error rate).

All the experiments reported in the paper were performed using GMTK (Bilmes and Zweig, 2002), a general purpose graphical model toolkit and SRILM (Stolcke, 2002), a language modeling toolkit.

Table 1 is populated with the results on the test set for the DBN-based frame composition systems in terms of precision, recall and F-measure. For the FE figures, only the reference FEs corresponding to correctly identified frames are considered. Only the frame and FE names are considered, neither their constituents nor their order matter. Finally, results are given for the sub-frame links between frames and FEs. Table 1 shows that the performances of the 3 DBN-based systems are quite comparable. Anyhow the 2-level system can be considered the best as besides its good F-measure results, it is also the most efficient model in terms of decoding complexity. The good results obtained for the sub-frame links confirm that the DBN models combined with a small rule set can be used to generate consistent hierarchical structures. Moreover, as they can provide hypotheses with confidence scores they can be used in a multiple input/output context (lattices and n-best lists) or in a validation process (evaluating and ranking hypotheses from other systems).

5 Conclusion

This work investigates a stochastic process for generating and composing semantic frames using dynamic Bayesian networks. The proposed approach offers a convenient way to automatically derive semantic annotations of speech utterances based on a complete frame and frame element hierarchical structure. Experimental results, obtained on the MEDIA dialog corpus, show that the performance of the DBN-based models are definitely good enough to be used in a dialog system in order to supply the dialog manager with a rich and thorough representation of the user's request semantics. Though this can also be obtained using a rule-based approach, the DBN models alone are able to derive n-best lists of semantic tree hypotheses with confidence scores. The incidence of such outputs on the dialog manager decision accuracy needs to be asserted.

Acknowledgment

This work is supported by the 6th Framework Research Program of the European Union (EU), LUNA Project, IST contract no 33549,www.ist-luna.eu

References

J. Bilmes and K. Kirchhoff. 2003. Factored language models and generalized parallel backoff. In *NAACL HLT*.

J. Bilmes and G. Zweig. 2002. The graphical models toolkit: An open source software system for speech and time-series processing. In *IEEE ICASSP*.

H. Bonneau-Maynard, S. Rosset, C. Ayache, A. Kuhn, D. Mostefa, and the Media consortium. 2005. Semantic annotation of the MEDIA corpus for spoken dialog. In *ISCA Eurospeech*.

C.J. Fillmore, C.R. Johnson, and M.R.L. Petruck. 2003. Background to framenet. *International Journal of Lexicography*, 16.3:235–250.

Y. He and S. Young. 2005. Spoken language understanding using the hidden vector state model. *Speech Communication*, 48(3-4):262–275.

F. Lefèvre. 2007. Dynamic bayesian networks and discriminative classifiers for multi-stage semantic interpretation. In *IEEE ICASSP*.

J.B. Lowe, C.F. Baker, and C.J. Fillmore. 1997. A frame-semantic approach to semantic annotation. In *SIGLEX Workshop: Why, What, and How?*

A. Stolcke. 2002. Srilm an extensible language modeling toolkit. In *IEEE ICASSP*.

Evaluation of a System for Noun Concepts Acquisition from Utterances about Images (SINCA) Using Daily Conversation Data

Yuzu UCHIDA
Graduate School of
Information Science and Technology
Hokkaido University
Sapporo, 060-0814, Japan
yuzu@media.eng.hokudai.ac.jp

Kenji ARAKI
Graduate School of
Information Science and Technology
Hokkaido University
Sapporo, 060-0814, Japan
araki@media.eng.hokudai.ac.jp

Abstract

For a robot working in an open environment, a task-oriented language capability will not be sufficient. In order to adapt to the environment, such a robot will have to learn language dynamically. We developed a System for Noun Concepts Acquisition from utterances about Images, SINCA in short. It is a language acquisition system without knowledge of grammar and vocabulary, which learns noun concepts from user utterances. We recorded a video of a child's daily life to collect dialogue data that was spoken to and around him. The child is a member of a family consisting of the parents and his sister. We evaluated the performance of SINCA using the collected data. In this paper, we describe the algorithms of SINCA and an evaluation experiment. We work on Japanese language acquisition, however our method can easily be adapted to other languages.

1 Introduction

There are several other studies about language acquisition systems. Rogers et al. (1997) proposed "Babbette", which learns language rules from provided examples. Levinson et al. (2005) describe their research with a robot which acquires language from interaction with the real world. Kobayashi et al. (2002) proposed a model for child vocabulary acquisition based on an inductive logic programming framework. Thompson (1995) presented a lexical acquisition system that learns a mapping of words to their semantic representation from training examples consisting of sentences paired with their semantic representations.

As mentioned above, researchers are interested in making a robot learn language. Most studies seem to be lacking in the ability to adapt to the real world. In addition, they should be more independent from language rules. We believe that it is necessary to simulate human language ability in order to create a complete natural language understanding system.

As the first step in our research, we developed a System for Noun Concepts Acquisition from utterances about Images, called SINCA in short (which means "evolution" in Japanese) (Uchida et al., 2007). It is a language acquisition system without knowledge of grammar and vocabulary, which learns noun concepts from a user's input. SINCA uses images as a meaning representation in order to eliminate ambiguity of language. SINCA can only acquire concrete nouns.

Currently, SINCA is for Japanese only. The language acquisition method of this system is very general and it is independent of language rules. SINCA is expected to work successfully using any language.

In this paper, we describe the algorithms of SINCA and an experiment to test what kind of input would be appropriate for our system. We would emphasize that we prepared a large video data of daily life of a family with young children.

2 The Algorithms of SINCA

Figure 1 shows the SINCA user interface. The situation shown in Fig.1 is that the affection of SINCA is directed to an eraser by the user, and after the recognition process, SINCA asks *"KESHIGOMU?*

Proceedings of NAACL HLT 2009: Short Papers, pages 65–68,
Boulder, Colorado, June 2009.

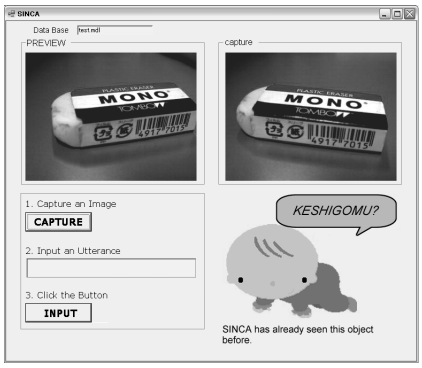


Figure 1: The SINCA Interface recognizing an eraser

*Kore-ha **KAPPU**-tte iu-n-da-yo.*
(This is a thing called a cup.)
***KAPPU**-ni gyūnyū ireyōka.*
(Let's pour some milk into the cup.)

Strings indicated by boldface are labels.

Figure 2: Examples of input data

(Eraser?)."

We describe SINCA's process in detail in the following subsections.

2.1 Input

A user input consists of an image paired with a spoken utterance.

First, a user chooses an object *O* which he or she likes and captures an image of it with a web camera with 300,000 pixels effective sensor resolution. The user has to try to capture the whole object *O* in the image.

Next, a user imagines an utterance that an infant might be exposed to when listening to caregivers while gazing at the object *O* in the environment. The user enters the utterance on the keyboard as a linguistic input. The linguistic input is written in *Hiragana*, which are Japanese phonetic characters, to avoid the linguistic input containing some direct meanings as in the case of Chinese *Kanji* ideograms. This is also intended to standardize the transcription. SINCA does not carry out morphological analysis of the linguistic input, because we believe that infant capability for word segmentation is not perfect (Jusczyk et al., 1999).

Figure 2 shows some example inputs. [1]

2.2 Image Processing

The ERSP 3.1 Software Development Kit [2] provides cutting edge technologies for vision, navigation, and system development. ERSP Vision included in the ERSP enables a robot or device to recognize 2D and 3D objects in real world settings where lighting and placement are not controlled. We use the ERSP vision for image processing. ERSP Vision informs the system whether the object in the present input image appears in the previously input images or not.

2.3 Common Parts

When a user inputs an image of an object *O* and an utterance, the system extracts all sections of the string matching section of previously input utterances accompanied by the image of the same object *O*. We call these strings common parts. After this process, the system deals with them as candidates for a label for the object *O*.

The system provides every common part with a "basic score". The basic score is based on frequency of appearance and the number of characters, and indicates how appropriate as a label the common part is. The higher the score, the more appropriate the common part is. The basic score is defined as follows:

$$SCORE = \alpha \times \frac{F}{PN} \times \sqrt{L} \qquad (1)$$

where, α is a coefficient which reduces the basic score if the common part has appeared with other objects than *O*, F is frequency of appearance of the common part with the images of *O*, PN is the number of use inputs with images of *O*, and L is the number of characters of the common part.

[1] The Japanese words are written in italics in all following figures.

[2] Evolution Robotics, Inc.:ERSP 3.1 Robotic Development Platform OEM Software by Evolution Robotics

2.4 Output

If the system finds a common part whose basic score exceeds a threshold, it outputs it as text. The reason for doing this is the assumption that there is a high possibility that such common parts are appropriate as labels.

A user evaluates an output by choosing one of the following keywords:

- Good : It is appropriate as a label.
- Almost : It makes some sense but is not proper for the label.
- Bad : It makes no sense.

Infants cannot understand these keywords completely, but they can get a sense of some meanings from the tone of an adult's voice or facial expressions. In our research, we use the keywords as a substitute for such information. The system recalculates the basic score based on the keyword chosen by the user. Specifically, the system multiplies the basic score by the coefficient β dependent on the keyword.

2.5 Acquisition of the Noun Concepts

After repeating these processes, if there is a common part whose score is more than 30.0 and which has been rated as "Good", the system acquires the common part as the label for *O*.

2.6 Label Acquisition Rules

Humans can use their newfound knowledge to learn their native language effectively. This system imitates humans' way with "label acquisition rules".

A label acquisition rule is like a template, which enables recursive learning for acquisition of noun concepts. The system generates label acquisition rules after acquisition of a label. When the system acquires a string S as a label for an object, the system picks up the previous linguistic inputs with the images of the object which contain the string S. Then, the system replaces the string S in the linguistic inputs with a variable "γ". These abstracted sentences are called label acquisition rules. An example of the label acquisition rules is shown in Fig.3.

If the rules match other parts of previoiusly input strings, the parts corresponding to the "γ" variable are extracted. The scores of these extracted strings are then increased.

Acquired Label	: ***WAN-CHAN*** (a doggy)
Previous Input	: *Acchi-ni* ***WAN-CHAN****-ga iru-yo.* (There is a doggy over there.)
Label Acquisition Rule	: *Acchi-ni* γ***1****-ga iru-yo.* (There is γ1 over there.)

Strings indicated by boldface are labels.

Figure 3: An example of a label acquisition rule

3 Evaluation Experiment

We carried out an experiment to test what kinds of input would be appropriate for SINCA. This section describes the experiment.

3.1 Experimental Procedure

Two types of linguistic input data were collected in two different ways: a questionnaire and a video recording. We had SINCA acquire labels for 10 images using the linguistic input data. The following are the details about the data collection methods.

3.1.1 Questionnaire

10 images were printed on the questionnaire, and it asked "What would you say to a young child if he or she pays attention to these objects?". The respondents are allowed to answer with whatever they come up with. 31 people responded to this questionnaire, and 13 of them have children of their own. We collected 324 sentences, and the average mora length of them was 11.0.

3.1.2 Video recording

We recorded a video of a child's daily life to collect dialogue data that was spoken to and around him. The child is a member of a family consisting of his parents and his sister.

The recordings are intended to collect daily conversation, therefore we did not set any tasks. The total recording period comprised 125 days and we recorded about 82 hours of video data. The first author watched about 26 hours of the video data, and wrote parents' dictation in *Hiragana*. We selected 353 sentences for linguistic input data that were spoken when joint attention interactions between a parent and a child were recognized. On average, their mora length was 9.8.

3.2 Experimental Result

We input sentences from the collected inputs one at a time until SINCA acquired a noun concept for an image. SINCA was able to acquire labels for 10 images, with each type of linguistic input. When we used the questionnaire data, SINCA needed on average 6.2 inputs to acquire one label, and SINCA acquired 52 rules through the experiment. They cover 83.8% of the total number of inputs. When we used the video data, SINCA needed on average 5.3 inputs to acquire one label, and SINCA acquired 44 rules through the experiment. They cover 83.0% of the total number of inputs.

3.3 Considerations

The experimental results indicate that using video data makes the acquisition of labels more efficient. There are 3 factors that contribute to this.

The first factor is the number of one-word sentences. There are 66 one-word sentences in the video data (18.6% of the total). Therefore, the length of the sentences from the video data tends to be short.

The second factor is the lack of particles. The respondents of the questionnaire hardly ever omit particles. By contrast, of the 53 sentences which were input, 23 sentences lack particles (42.6% of the total) in video data. Spoken language is more likely to have omitted particles compared with written language.

The third factor is the variety of words. We randomly selected 100 sentences from both sets of linguistic input data and checked the words adjacent to a label. Table 1 shows the number of different words that occur adjacent to a label. Because the respondents of the questionnaire all try to explain something in an image, they use similar expressions.

When SINCA uses the video data, it can extract labels more easily than using the questionnaire data because of the factors listed above. This means that SINCA is well suited for spoken language. If we assume one application of SINCA is for communication robots, this result is promising.

Table 1: Variety of words

	Previous(W_A)	following(W_B)
Video	19	42
Questionnaire	15	22

$Sentence : W_1\ W_2\ ...\ W_A$ label $W_B\ ...$.

4 Conclusions and Future Work

In this paper, we described the algorithms of SINCA. SINCA can acquire labels for images without ready-made linguistic resources, lexical information, or syntactic rules. Additionally, it targets images of real world objects.

We collected linguistic input data in two ways. One method is videos of a family's daily life. The other method is a questionnaire. We had SINCA acquire noun concepts using both video and questionnaire data. As a result, we have showed that spoken language is well suited to SINCA's algorithm for acquiring noun concepts.

In the next step, we will focus on acquisition of adjectives.

References

Jusczyk, P. W. Houston, D. M. and Newsome, M. 1999. *The beginnings of word segmentation in english-learning infants*. Cognitive Psychology. **39**. pp.159–207.

Kobayashi, I. Furukawa, K. Ozaki, T. and Imai, M. 2002. *A Computational Model for Children's Language Acquisition Using Inductive Logic Programming*. Progress in Discovery Science. **2281** pp.140–155.

Levinson S. E. Squire, K. Lin, R. S. and McClain, M. 2005. *Automatic language acquisition by an autonomous robot*. AAAI Spring Symposium on Developmental Robotics.

Rogers, P. A. P. and Lefley, M. 1997. *The baby project*. Machine Conversations. ed. Wilks, Y. Kluwer Academic Publishers.

Thompson, C, A. 1997. *Acquisition of a Lexicon from Semantic Representations of Sentences*. Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics. pp.335–337.

Uchida, Y. and Araki, K. 2007. *A System for Acquisition of Noun Concepts from Utterances for Images Using the Label Acquisition Rules*. Springer-Verlag Lecture Notes in Artificial Intelligence (LNAI). pp.798–802.

Web and Corpus Methods for Malay Count Classifier Prediction

Jeremy Nicholson and **Timothy Baldwin**
NICTA Victoria Research Laboratories
University of Melbourne, VIC 3010, Australia
{jeremymn,tim}@csse.unimelb.edu.au

Abstract

We examine the capacity of Web and corpus frequency methods to predict preferred count classifiers for nouns in Malay. The observed F-score for the Web model of 0.671 considerably outperformed corpus-based frequency and machine learning models. We expect that this is a fruitful extension for Web–as–corpus approaches to lexicons in languages other than English, but further research is required in other South-East and East Asian languages.

1 Introduction

The objective of this paper is to extend a Malay lexicon with count classifier information for nominal types. This is done under the umbrella of deep lexical acquisition: the process of automatically or semi-automatically learning linguistic structures for use in linguistically rich language resources such as precision grammars or wordnets (Baldwin, 2007).

One might call Malay a "medium-density" language: some NLP resources exist, but substantially fewer than those for English, and they tend to be of low complexity. Resources like the Web seem promising for bootstrapping further resources, aided in part by simple syntax and a Romanised orthographic system. The vast size of the Web has been demonstrated to combat the data sparseness problem, for example, in Lapata and Keller (2004).

We examine using a similar "first gloss" strategy to Lapata and Keller (akin to "first sense" in WSD, in this case, identifying the most basic surface form that a speaker would use to disambiguate between possible classes), where the Web is used a corpus to query a set of candidate surface forms, and the frequencies are used to disambiguate the lexical property. Due to the heterogeneity of the Web, we expect to observe a significant amount of blocking from Indonesian, a language with which Malay is somewhat mutually intelligible (Gordon, 2005). Hence, we contrast this approach with observing the cues directly from a corpus strictly of Malay, as well as a corpus-based supervised machine learning approach which does not rely on a presupplied gloss.

2 Background

2.1 Count Classifiers

A count classifier (CL) is a noun that occurs in a specifier phrase with one of a set of (usually numeric) specifiers; the specifier phrase typically occurs in apposition or as a genitive modifier (GEN) to the head noun. In many languages, including many South-East Asian, East Asian, and African families, almost all nouns are uncountable and can only be counted through specifier phrases. A Malay example, where *biji* is the count classifier (CL) for fruit, is given in (1).

(1) *tiga* *biji* *pisang*
three CL banana
"three bananas"

Semantically, a lexical entry for a noun will include a default (sortal) count classifier which selects for a particular semantic property of the lemma. Usually this is a conceptual class (e.g. HUMAN or ANIMAL) or a description of some relative dimensional property (e.g. FLAT or LONG-AND-THIN).

Since each count classifier has a precise semantics, using a classifier other than the default can coerce a given lemma into different semantics. For example, *raja* "king" typically takes *orang* "person" as a classifier, as in *2 orang raja* "2 kings", but can take on an animal reading with *ekor* "animal" in *2 ekor raja* "2 kingfishers". An unintended classifier

Proceedings of NAACL HLT 2009: Short Papers, pages 69–72,
Boulder, Colorado, June 2009.

can lead to highly marked or infelicitious readings, such as *#2 biji raja* "2 (chess) kings".

Most research on count classifiers tends to discuss generating a hierarchy or taxonomy of the classifiers available in a given language (e.g. Bond and Paik (1997) for Japanese and Korean, or Shirai et al. (2008) cross-linguistically) or using language-specific knowledge to predict tokens (e.g. Bond and Paik (2000)) or both (e.g. Sornlertlamvanich et al. (1994)).

2.2 Malay Data

Little work has been done on NLP for Malay, however, a stemmer (Adriani et al., 2007) and a probabilistic parser for Indonesian (Gusmita and Manurung, 2008) have been developed. The mutually intelligibility suggests that Malay resources could presumably be extended from these.

In our experiments, we make use of a Malay–English translation dictionary, KAMI (Quah et al., 2001), which annotates about 19K nominal lexical entries for count classifiers. To limit very low frequency entries, we cross-reference these with a corpus of 1.2M tokens of Malay text, described in Baldwin and Awab (2006). We further exclude the two non-sortal count classifiers that are attested as default classifiers in the lexicon, as their distribution is heavily skewed and not lexicalised.

In all, 2764 simplex common nouns are attested at least once in the corpus data. We observe 2984 unique noun–to–default classifier assignments. Polysemy leads to an average of 1.08 count classifiers assigned to a given wordform. The most difficult exemplars to classify, and consequently the most interesting ones, correspond to the dispreferred count classifiers of the multi-class wordforms: direct assignment and frequency thresholding was observed to perform poorly. Since this task is functionally equivalent to the subcat learning problem, strategies from that field might prove helpful (e.g. Korhonen (2002)).

The final distribution of the most frequent classes is as follows:

CL:	*orang*	*buah*	*batang*	*ekor*	OTHER
Freq:	0.389	0.292	0.092	0.078	0.149

Of the 49 classes, only four have a relative frequency greater than 3% of the types: *orang* for people, *batang* for long, thin objects, *ekor* for animals, and *buah*, the semantically empty classifier, for when no other classifiers are suitable (e.g. for abstract nouns); *orang* and *buah* account for almost 70% of the types.

3 Experiment

3.1 Methodology

Lapata and Keller (2004) look at a set of generation and analysis tasks in English, identify simple surface cues, and query a Web search engine to approximate those frequencies. They then use maximum likelihood estimation or a variety of normalisation methods to choose an output.

For a given Malay noun, we attempt to select the default count classifier, which is a generation task under their framework, and semantically most similar to noun countability detection. Specifier phrases almost always premodify nouns in Malay, so the set of surface cues we chose was *satu* CL NOUN "one/a NOUN".[1] This was observed to have greater coverage than *dua* "two" and other non-numeral specifiers. 49 queries were performed for each headword, and maximum likelihood estimation was used to select the predicted classifier (i.e. taking most frequently observed cue, with a threshold of 0). Frequencies from the same cues were also obtained from the corpus of Baldwin and Awab (2006).

We contrasted this with a machine learning model for Malay classifiers, designed to be language-independent (Nicholson and Baldwin, 2008). A feature vector is constructed for each headword by concatenating context windows of four tokens to the left and right of each instance of the headword in the corpus (for eight word unigram features per instance). These are then passed into two kinds of maximum entropy model: one conditioned on all 49 classes, and one cascaded into a suite of 49 separate binary classifiers designed to predict each class separately. Evaluation is via 10-fold stratified cross-validation. A majority class baseline was also examined, where every headword was assigned the *orang* class.

For the corpus-based methods, if the frequency of every cue is 0, no prediction of classifier is made. Similarly, the suite can predict a negative assign-

[1] *satu* becomes cliticised to *se-* in this construction, so that instead of cues like *satu buah raja, satu orang raja, ...*, we have cues like *sebuah raja, seorang raja, ...*

Method	Web	Corpus	Suite	Entire	Base
Prec.	.736	.908	.652	.570	.420
Rec.	.616	.119	.379	.548	.389
$F_{\beta}=1$	.671	.210	.479	.559	.404

Table 1: Performance of the five systems.

Back-off	Web	Suite	Entire	*orang*	*buah*
Prec.	.736	.671	.586	.476	.389
Rec.	.616	.421	.561	.441	.360
$F_{\beta}=1$	.671	.517	.573	.458	.374

Table 2: Performance of corpus frequency assignment (Corpus in Table 1), backed-off to the other systems.

ment for each of the 49 classes. Consequently, precision is calculated as the fraction of correctly predicted instances to the number of examplars where a prediction was made. Only the suite of classifiers could natively handle multi-assignment of classes: recall was calculated as the fraction of correctly predicted instances to all 2984 possible headword–class assignments, despite the fact that four of the systems could not make 220 of the classifications.

3.2 Results

The observed precision, recall, and F-scores of the various systems are shown in Table 1. The best F-score is observed for the Web frequency system, which also had the highest recall. The best precision was observed for the corpus frequency system, but with very low recall — about 85% of the wordforms could not be assigned to a class (the corresponding figure for the Web system was about 9%). Consequently, we attempted a number of back-off strategies so as to improve the recall of this system.

The results for backing off the corpus frequency system to the Web model, the two maximum entropy models, and two baselines (the majority class, and the semantically empty classifier) are shown in Table 2. Using a Web back-off was nearly identical to the basic Web system: most of the correct assignments being made by the corpus frequency system were also being captured through Web frequencies, which indicates that these are the easier, high frequency entries. Backing off to the machine learning models performed the same or slightly better than using the machine learning model by itself. It therefore seems that the most balanced corpus-based model should take this approach.

The fact that the Web frequency system had the best performance belies the "noisiness" of the Web, in that one expects to observe errors caused by carelessness, laziness (e.g. using *buah* despite a more specific classifier being available), or noise (e.g. Indonesian count classifier attestation; more on this below). While the corpus of "clean", hand-constructed data did have a precision improvement over the Web system, the back-off demonstrates that it was not substantially better over those entries that could be classified from the corpus data.

4 Discussion

As with many classification tasks, the Web-based model notably outperformed the corpus-based models when used to predict count classifiers of Malay noun types, particularly in recall. In a type-wise lexicon, precision is probably the more salient evaluation metric, as recall is more meaningful on tokens, and a low-precision lexicon is often of little utility; the Web system had at least comparable precision for the entries able to be classified by the corpus-based systems.

We expected that the heterogeneity of the Web, particularly confusion caused by a preponderance of Indonesian, would cause performance to drop, but this was not the case. The Ethnologue estimates that there are more speakers of Indonesian than Malay (Gordon, 2005), and one would expect the Web distribution to reflect this. Also, there are systematic differences in the way count classifiers are used in the two languages, despite the intelligibility; compare "five photographs": *lima keping foto* in Malay and *lima lembar foto, lima foto* in Indonesian.

While the use of count classifiers is obligatory in Malay, it is optional in Indonesian for lower registers. Also, many classifiers that are available in Malay are not used in Indonesian, and the small set of Indonesian count classifiers that are not used in Malay do not form part of the query set, so no confusion results. Consequently, it seems that greater difficulty would arise when attempting to predict count classifiers for Indonesian nouns, as their optionality and blocking from Malay cognates would introduce noise in cases where language identification has not been used to generate the corpus (like the

Web) — hand-constructed corpora might be necessary in that case. Furthermore, the Web system benefits from a very simple surface form, namely *se-*CL NOUN: languages that permit floating quantification, like Japanese, or require classifiers for stative verb modification, like Thai, would need many more queries or lower-precision queries to capture most of the cues available from the corpus. We intend to examine these phenomena in future work.

An important contrast is noted between the "unsupervised" methods of the corpus-frequency systems and the "supervised" machine learning methods. One presumed advantage of unsupervised systems is the lack of pre-annotated training data required. In this case, a comparable time investment by a lexicographer would be required to generate the set of surface forms for the corpus-frequency models. The performance dictates that the glosses for the Web system give the most value for lexicographer input; however, for other languages or other lexical properties, generating a set of high-precision, high-recall glosses is often non-trivial. If the Web is not used, having both training data and high-precision, low-recall glosses is valuable.

5 Conclusion

We examine an approach for using Web and corpus data to predict the preferred generation form for counting nouns in Malay, and observed greater precision than machine learning methods that do not require a presupplied gloss. Most Web–as–corpus research tends to focus on English; as the Web increases in multilinguality, it becomes an important resource for medium- and low-density languages. This task was quite simple, with glosses amenable to Web approaches, and is promising for automatically extending the coverage of a Malay lexicon. However, we expect that the Malay glosses will block readings of Indonesian classifiers, and classifiers in other languages will require different strategies; we intend to examine this in future work.

Acknowledgements

We would like to thank Francis Bond for his valuable input on this research. NICTA is funded by the Australian government as represented by Department of Broadband, Communication and Digital Economy, and the Australian Research Council through the ICT Centre of Excellence programme.

References

M. Adriani, J. Asian, B. Nazief, S.M.M. Tahaghoghi, and H.E. Williams. 2007. Stemming Indonesian: A confix-stripping approach. *ACM Transactions on Asian Language Information Processing*, 6.

T. Baldwin and S. Awab. 2006. Open source corpus analysis tools for Malay. In *Proc. of the 5th International Conference on Language Resources and Evaluation*, pages 2212–5, Genoa, Italy.

T. Baldwin. 2007. Scalable deep linguistic processing: Mind the lexical gap. In *Proc. of the 21st Pacific Asia Conference on Language, Information and Computation*, pages 3–12, Seoul, Korea.

F. Bond and K. Paik. 1997. Classifying correspondence in Japanese and Korean. In *Proc. of the 3rd Conference of the Pacific Association for Computational Linguistics*, pages 58–67, Tokyo, Japan.

F. Bond and K. Paik. 2000. Reusing an ontology to generate numeral classifiers. In *Proc. of the 19th International Conference on Computational Linguistics*, pages 90–96, Saarbrücken, Germany.

R.G. Gordon, Jr, editor. 2005. *Ethnologue: Languages of the World, Fifteenth Edition*. SIL International.

R.H. Gusmita and Ruli Manurung. 2008. Some initial experiments with Indonesian probabilistic parsing. In *Proc. of the 2nd International MALINDO Workshop*, Cyberjaya, Malaysia.

A. Korhonen. 2002. *Subcategorization Acquisition*. Ph.D. thesis, University of Cambridge, Cambridge, UK.

M. Lapata and F. Keller. 2004. The web as a baseline: Evaluating the performance of unsupervised web-based models for a range of NLP tasks. In *Proc. of the 4th International Conference on Human Language Technology Research and 5th Annual Meeting of the NAACL*, pages 121–128, Boston, USA.

J. Nicholson and T. Baldwin. 2008. Learning count classifier preferences of Malay nouns. In *Proc. of the Australasian Language Technology Association Workshop*, pages 115–123, Hobart, Australia.

C.K. Quah, F. Bond, and T. Yamazaki. 2001. Design and construction of a machine-tractable Malay-English lexicon. In *Proc. of the 2nd Biennial Conference of ASIALEX*, pages 200–205, Seoul, Korea.

K. Shirai, T. Tokunaga, C-R. Huang, S-K. Hsieh, T-Y. Kuo, V. Sornlertlamvanich, and T. Charoenporn. 2008. Constructing taxonomy of numerative classifiers for Asian languages. In *Proc. of the Third International Joint Conference on Natural Language Processing*, Hyderabad, India.

V. Sornlertlamvanich, W. Pantachat, and S. Meknavin. 1994. Classifier assignment by corpus-based approach. In *Proc. of the 15th International Conference on Computational Linguistics*, pages 556–561, Kyoto, Japan.

Minimum Bayes Risk Combination of Translation Hypotheses from Alternative Morphological Decompositions

Adrià de Gispert[‡] **Sami Virpioja**[⋆] **Mikko Kurimo**[⋆] **William Byrne**[‡]
[‡] University of Cambridge. Dept. of Engineering. CB2 1PZ Cambridge, U.K.
{ad465,wjb31}@eng.cam.ac.uk
[⋆] Helsinki University of Technology. Adaptive Informatics Research Centre
P.O.Box 5400, 02015 TKK, Finland
{sami.virpioja,mikko.kurimo}@tkk.fi

Abstract

We describe a simple strategy to achieve translation performance improvements by combining output from identical statistical machine translation systems trained on alternative morphological decompositions of the source language. Combination is done by means of Minimum Bayes Risk decoding over a shared N-best list. When translating into English from two highly inflected languages such as Arabic and Finnish we obtain significant improvements over simply selecting the best morphological decomposition.

1 Introduction

Morphologically rich languages pose significant challenges for natural language processing. The extensive use of inflection, derivation, and composition leads to a huge vocabulary, and sparsity in models estimated from data. Statistical machine translation (SMT) systems estimated from parallel text are affected by this. This is particularly acute when either the source or the target language, or both, are morphologically complex.

Owing to these difficulties and to the natural interest researchers take in complex linguistic phenomena, many approaches to morphological analysis have been developed and evaluated. We focus on applications to SMT in Section 1.1, but we note the recent general survey (Roark and Sproat, 2007) and the Morpho Challenge competitive evaluations[1]. Prior evaluations of morphological analyzers have focused on determining which analyzer was best suited for some particular task. For translation, we take a different approach and investigate whether competing analyzers might have complementary information. Our method is straightforward. We train two identical SMT systems with two versions of the same parallel corpus, each with a different morphological decomposition of the source language. We combine their translation hypotheses performing Minimum Bayes Risk decoding over merged N-best lists. Results are reported in the NIST 2008 Arabic-to-English MT task and an European Parliament Finnish-to-English task, with significant gains over each individual system.

1.1 Prior Work

Several earlier works investigate word segmentation and transformation schemes, which may include Part-Of-Speech or other information, to alleviate the effect of morphological variation on translation models. With different training corpus sizes, they focus on translation *into* English from Arabic (Lee, 2004; Habash and Sadat, 2006; Zollmann et al., 2006), Czech (Goldwater and McClosky, 2005; Talbot and Osborne, 2006), German (Nießen and Ney, 2004) or Catalan, Spanish and Serbian (Popovic and Ney, 2004). Some address the generation challenge when translating *from* English into Spanish (Ueffing and Ney, 2003; de Gispert and Mariño, 2008). Unsupervised morphology learning is proposed as a language-independent solution to reduce the problems of rich morphology in (Virpioja et al.,

[1] See http://www.cis.hut.fi/morphochallenge2009/ and links there to earlier workshops. The combination scheme described in this paper will be one of the evaluation tracks in the upcoming workshop.

Proceedings of NAACL HLT 2009: Short Papers, pages 73–76,
Boulder, Colorado, June 2009.

Arabic	wqrrt An tn$A ljnp tHDyryp jAmEp lljmEyp AlEAmp fY dwrthA AlvAnyp wAlxmsyn
MADA D2	w+ qrrt >n tn$A ljnp tHDyryp jAmEp l+ AljmEyp AlEAmp fy dwrthA AlvAnyp w+ Alxmsyn
SAKHR	w+ qrrt An tn$A ljnp tHDyryp jAmEp l*l+ jmEyp Al+ EAmp fY dwrt +hA Al+ vAnyp w*Al+ xmsyn
English	a preparatory committee of the whole of the general assembly is to be established at its fifty-second session

Table 1: Example of alternative segmentation schemes for a given Arabic sentence, in Buckwalter transliteration.

2007). Factored models are introduced in (Koehn and Hoang, 2007) for better integration of morpho-syntactic information.

Giménez and Màrquez (2005) merge multiple word alignments obtained from several linguistically-tagged versions of a Spanish-English corpus, but only standard tokens are used in decoding. Dyer et al. (2008) report improvements from multiple Arabic segmentations in translation to English translation, but their goal was to demonstrate the value of lattice-based translation. From a modeling perspective their approach is unwieldy: multiple analyses of the parallel text collections are merged to create a large, heterogeneous training set; a single set of models and alignments is produced; lattice translation is then performed using a single system to translate all morphological analyses. We find that similar gains can be obtained much more easily.

The approach we take is Minimum Bayes Risk (MBR) System Combination (Sim et al., 2007). N-best lists from multiple SMT systems are merged; the posterior distributions over the individual lists are interpolated to form a new distribution over the merged list. MBR hypotheses selection is then performed using sentence-level BLEU score (Kumar and Byrne, 2004). It is very likely that even greater gains can be achieved by more complicated combination schemes (Rosti et al., 2007), although significantly more effort in tuning would be required.

2 Arabic-to-English Translation

For Arabic-to-English translation, we consider two alternative segmentations of the Arabic words. We first use the MADA toolkit (Habash and Rambow, 2005). After tagging, we split word prefixes and suffixes according to scheme 'D2' (Habash and Sadat, 2006). Secondly, we take the segmentation generated by Sakhr Software in Egypt using their Arabic Morphological Tagger, as an alternative segmentation into subword units. This scheme generates more tokens as it segments all Arabic articles which otherwise remain attached in the MADA D2 scheme (Table 1).

Translation experiments are based on the NIST MT08 Arabic-to-English translation task, including all allowed parallel data as training material ($\sim$150M English words, and 153M or 178M Arabic words for MADA-segmented and Sakhr-segmented text, respectively). In addition to the MT08 set itself, we take the NIST MT02 through MT05 evaluation sets and divide them into a development set (odd-numbered sentences) and a test set (even-numbered sentences), each containing $\sim$2k sentences.

The SMT system used is *HiFST*, a hierarchical phrase-based system implemented with Weighted Finite-State Transducers (Iglesias et al., 2009). Two identical systems are trained from each parallel corpus, i.e. MADA-based and SAKHR-based. Both systems use the same standard features and share the first-pass English language model, a 4-gram estimated over the parallel text and a 965 million word subset of monolingual data from the English Gigaword Third Edition. Minimum Error Training parameter estimation under IBM BLEU is performed on the development set (mt02-05-tune), and the output translation lattice is rescored with large language models estimated using $\sim$4.7B words of English newswire text, in the same fashion as (Iglesias et al., 2009). Finally, the first 1000-best hypotheses are rescored with MBR, taking the negative sentence level BLEU score as the loss function to minimise.

For system combination, we obtain two sets of N-best lists of depth N=500, one from each system. Both lists are obtained after large-LM lattice rescoring, i.e. prior to individual MBR. A joint MBR decoding is then carried out on the aggregated 1000-best list with equal weight assigned to the posterior distribution assigned to the hypotheses by each system. Results are shown in Table 2.

As shown, the scores obtained via MBR combination outperform significantly those achieved via MBR for the best-performing system (MADA). The

	mt02-05-		
	-tune	-test	mt08
MADA-based	53.3	52.7	43.7
+MBR	53.7	53.3	44.0
SAKHR-based	52.7	52.8	43.3
+MBR	53.2	53.2	43.8
MBR-combined	54.6	54.6	45.6

Table 2: Arabic-to-English translation results. Lowercased IBM BLEU reported.

mixed case BLEU-4 for the MBR-combined system on *mt08* is 44.1. This is directly comparable to the official MT08 Constrained Training Track evaluation results.[2]

3 Finnish-to-English Translation

Finnish is a highly-inflecting, agglutinative language. It has dozens of both inflectional and derivational suffixes, that are concatenated together with only moderately small changes in the surface forms. For instance, one can inflect the word ”kauppa” (shop) into ”kaupa+ssa+mme+kin” (also in our shop) by glueing the suffixes to the end. In addition, Finnish has many compound words, sometimes consisting of several parts, such as ”ulko+maa+n+kauppa+politiikka” (foreign trade policy). Due to these properties, the number of different word forms that can be observed is enormous.

Morfessor (Creutz and Lagus, 2007) is a method for modeling concatenative morphology in an unsupervised manner. It tries to find morpheme-like units, morphs, that are segments of the words. Inspired by the minimum description length principle, Morfessor tries to find a concise lexicon of morphs that can effectively code the words in the training data. Unlike other unsupervised methods (e.g., Goldsmith (2001)), there is no restrictions on how many morphs a word can have. After training the model, the most likely segmentation of new words to morphs can be found using the Viterbi algorithm.

There exist a few different versions of Morfessor. The baseline algorithm has been found to be very useful in automatic speech recognition of agglutinative languages (Kurimo et al., 2006). However, it often oversegments morphemes that are rare or not seen at all in the training data. Following the approach in (Virpioja et al., 2007), we use the Morfessor Categories-MAP algorithm (Creutz and Lagus, 2005). It applies a hierarchical model with three surface categories (prefix, stem and suffix), that allow the algorithm to treat out-of-vocabulary words in a convenient manner. For instance, if we encounter a new name with a known suffix, it can usually separate the suffix and leave the actual name intact.

Similarly to the Arabic-to-English task, we train two identical HiFST systems. In this case, whereas one is trained on Finnish morphs decomposed by Morfessor (morph-based), the other is trained on standard, unprocessed Finnish (word-based). For this task we use the EuParl parallel corpus . Portions from Q4/2000 was reserved for testing and September 2000 for development, both containing around 3,000 sentences. The training data comprised 23M English words, and 17M or 27M Finnish tokens for word-based or morph-based text, respectively.

The training set was also used to train the morphological segmentation. The quality of the segmentation is evaluated in (Virpioja et al., 2007). A precision of 78.72% and recall of 52.29% was measured for the segmentation boundaries with respect to a linguistic reference segmentation. As the recall is not very high, the segmentation is more conservative than the linguistic reference. Table 4 shows an example for a phrase in the training data.

Results are shown in Table 3, where again significant gains are achieved when simply combining output N-best lists via MBR. Only one reference was available for scoring. In this case we did not apply large-LM rescoring, as no large additional parliamentary data was available. Individual MBR did not yield gains for each of the systems.

	devel	test
Word-based	30.2	27.9
Morph-based	29.4	27.4
MBR-combined	30.5	28.9

Table 3: Finnish-to-English translation results. Lowercased IBM BLEU reported.

[2]Full MT08 results are available at http://www.nist.gov/speech/tests/mt/2008/doc/mt08_official_results_v0.html

Finnish	vaarallisten aineiden kuljetusten turvallisuusneuvonantaja
Morfessor	vaara$_{STM}$ llisten$_{STM}$ aine$_{STM}$ iden$_{SUF}$ kuljetus$_{PRE}$ ten$_{STM}$ turvallisuus$_{PRE}$ neuvo$_{STM}$ n$_{SUF}$ antaja$_{STM}$
Linguistic	vaara llis t en aine i den kuljet us t en turva llis uus neuvo n anta ja
English	safety adviser for the transport of dangerous goods

Table 4: Example of Morfessor Categories-MAP segmentation and linguistic segmentation for a Finnish phrase. Subscripts show the morph categories given by Morfessor: stem (STM), prefix (PRE) and suffix (SUF).

4 Conclusions

We demonstrated that multiple morphological analyses can be the basis for SMT system combination. These results will be of interest to researchers developing morphological analyzers, as it provides a new, and potentially profitable way to evaluate competing analysers. The results should also interest SMT researchers. SMT system combination is an active area of research, but good gains from combination usually require very different system architectures; this can be a barrier to developing competitive systems. We find that the same architecture trained on two different analyses is adequate to generate the diverse hypotheses needed for system combination.

Acknowledgments. This work was supported by the GALE program of DARPA (HR0011-06-C-0022), the GSLT and AIRC in the Academy of Finland, and the EMIME project and PASCAL2 NoE in the EC's FP7.

References

M. Creutz and K. Lagus. 2005. Inducing the morphological lexicon of a natural language from unannotated text. In *Conf. on Adaptive Knowledge Representation and Reasoning (AKRR)*.

M. Creutz and K. Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Trans. Speech and Language Processing*, 4(1).

A. de Gispert and J.B. Mariño. 2008. On the impact of morphology in English to Spanish statistical MT. *Speech Communication*, 50.

C. Dyer, S. Muresan, and P. Resnik. 2008. Generalizing word lattice translation. In *ACL-HLT*.

J. Giménez and Ll. Màrquez. 2005. Combining linguistic data views for phrase-based SMT. In *ACL Workshop on Building and Using Parallel Texts*.

J. Goldsmith. 2001. Unsupervised learning of the morphology of a natural language. *Computational Linguistics*, 27(2).

S. Goldwater and D. McClosky. 2005. Improving statistical MT through morphological analysis. In *HLT-EMNLP*.

N. Habash and O. Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *ACL*.

N. Habash and F. Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. In *HLT-NAACL: Short Papers*.

G. Iglesias, A. de Gispert, E.R. Banga, and W. Byrne. 2009. Hierarchical phrase-based translation with weighted finite state transducers. In *HLT-NAACL*.

P. Koehn and H. Hoang. 2007. Factored translation models. In *EMNLP*.

S. Kumar and W. Byrne. 2004. Minimum Bayes-risk decoding for statistical machine translation. In *HLT-NAACL*.

M. Kurimo, A. Puurula, E. Arisoy, V. Siivola, T. Hirsimäki, J. Pylkkönen, T. Alumäe, and M. Saraclar. 2006. Unlimited vocabulary speech recognition for agglutinative languages. In *HLT-NAACL*.

Y.-S. Lee. 2004. Morphological analysis for statistical machine translation. In *HLT-NAACL: Short Papers*.

S. Nießen and H. Ney. 2004. Statistical machine translation with scarce resources using morpho-syntactic information. *Computational Linguistics*, 30(2).

M. Popovic and H. Ney. 2004. Towards the use of word stems and suffixes for statistical machine translation. In *LREC*.

B. Roark and R. Sproat. 2007. *Computational Approaches to Morphology and Syntax*. Oxford University Press.

A.V. Rosti, S. Matsoukas, and R. Schwartz. 2007. Improved word-level system combination for machine translation. In *ACL*.

K.C. Sim, W. Byrne, M. Gales, H. Sahbi, and P. C. Woodland. 2007. Consensus network decoding for statistical machine translation system combination. In *ICASSP*, volume 4.

D. Talbot and M. Osborne. 2006. Modelling lexical redundancy for machine translation. In *ACL*.

N. Ueffing and H. Ney. 2003. Using POS information for SMT into morphologically rich languages. In *EACL*.

S. Virpioja, J.J. Väyrynen, M. Creutz, and M. Sadeniemi. 2007. Morphology-aware statistical machine translation based on morphs induced in an unsupervised manner. In *MT Summit XI*.

A. Zollmann, A. Venugopal, and S. Vogel. 2006. Bridging the inflection morphology gap for Arabic statistical machine translation. In *HLT-NAACL: Short Papers*.

Generating Synthetic Children's Acoustic Models from Adult Models

Andreas Hagen, Bryan Pellom, and Kadri Hacioglu

Rosetta Stone Labs
{ahagen, bpellom, khacioglu}@rosettastone.com

Abstract

This work focuses on generating children's HMM-based acoustic models for speech recognition from adult acoustic models. Collecting children's speech data is more costly compared to adult's speech. The patent-pending method developed in this work requires only adult data to estimate synthetic children's acoustic models in any language and works as follows: For a new language where only adult data is available, an adult male and an adult female model is trained. A linear transformation from each male HMM mean vector to its closest female mean vector is estimated. This transform is then scaled to a certain power and applied to the female model to obtain a synthetic children's model. In a pronunciation verification task the method yields 19% and 3.7% relative improvement on native English and Spanish children's data, respectively, compared to the best adult model. For Spanish data, the new model outperforms the available real children's data based model by 13% relative.

1 Introduction

Language learning is becoming more and more important in the age of globalization. Depending on their work or cultural situation some people are confronted with various different languages on a daily basis. While it is very desirable to learn languages at any age, language learning, among other learning experiences, is comparably simpler for children than for adults and should therefore be encouraged at early ages.

Even though the children's language learning market is highly important, comprising effective speech recognition tools for pronunciation assessment is relatively hard due to the special characteristics of children's speech and the limited availability of children's speech data in many languages in the speech research community. Adult speech data is usually easier to obtain. By understanding the characteristics of children's speech the unconditional need for children's speech data can be lessened by altering adult acoustic models such that they are suitable for children's speech.

Children's speech has higher pitch and formants than female speech. Further, female speech has higher pitch and formants than male speech. Children's speech is more variable than female speech, and, as research has shown, female speech is more variable than male speech (Lee et al., 1999). Given this transitive chain of argumentation, the transformation from a male to a female acoustic model can be estimated for a language and applied (at a certain adjustable degree) to the female model. This process results in a synthetic children's speech model designed on the basis of the female model. Therefore, for a new language an effective synthetic children's acoustic model can be derived without the need of children's data (Hagen et al., 2008).

2 Related Work

Extensive research has been done in the field of children's speech analysis and recognition in the past few years. A detailed overview of children's speech characteristics can be found in (Lee et al., 1999). The paper presents research results showing the higher variability in speech characteristics among children compared to adult speech. The properties of children's speech that were researched were duration of vowels and sentences, pitch, and formant locations.

When designing acoustic models specially suited for children, properties as the formant locations and higher variability of children's speech need to be accounted for. The best solution for building children's speech models is to collect children's speech data and to train models from scratch (Ha-

Proceedings of NAACL HLT 2009: Short Papers, pages 77–80,
Boulder, Colorado, June 2009.

gen et al., 2003, Cosi et al. 2005). Researchers have also tried to apply adult acoustic models using speaker normalization techniques to recognize children's speech (Elenius et al., 2005, Potamianos et al. 1997). Adult acoustic models were adapted towards children's speech. A limited amount of children's speech data was available for adaptation. In (Gustafson et al., 2002) children's voices were transformed before being sent to the recognizer using adult acoustic models. In (Claes et al., 1997) children's acoustic models were built based on a VTL adaptation of cepstral parameters based on the third formant frequency. The method showed to be effective for building children's speech models.

3 Building Synthetic Children's Models from Adult Models

As mentioned in Section 1, research has shown that pitch and formants of children's speech are higher than for female speech. Female speech has higher pitch and formants than male speech. In order to exploit these research results a transformation from a male acoustic model to a female acoustic model can be derived. This transformation will map a male model as close as possible to a female model. The transformation can be adjusted and applied to the female model. The resulting synthetic model can be tested on children's data.

Parameters that are subject to transformation in this process are the mean vectors of the HMM states. The transformation can be represented as a square matrix in the dimension of the mean vectors. The transformation chosen in this approach is therefore linear and is for example capable of representing a vocal tract length adaptation as it was shown in (Pitz et al., 2005). Linear transformations (i.e. matrices) are also chosen in adaptation approaches as MAPLR and MLLR, whose benefit has been shown to be additive to the benefit of VTLN in speaker adaptation applications. A linear transform in the form of a matrix is therefore well suited due to its expressive power as well as its mathematical manageability.

3.1 Transformation Matrix

The transformation matrix used in this approach is estimated by mapping the male to the female acoustic model, such that each HMM state mean vector in the male model is assigned a corresponding mean vector in the female model. Information used in the mapping process is the basic phoneme and context. The resulting mean vector pairs are used as source and target features in the training process of the transformation matrix. During training the matrix is initialized as the identity matrix and the estimate of the mapping is refined by gradient descent. In a typical acoustic model there are several hundred, sometimes thousands, of these mean vector pairs to train the transformation matrix. The expression that needs to be minimized is:

$$T = \arg\min_{A} \sum_{(x,y)\,pairs} (Ax - y)^2$$

where T is the error-minimizing transformation matrix; *x* is a male model's source vector and *y* it corresponding female model's target vector.

In this optimization process the Matrix *A* is initialized as the identity matrix. Each matrix entry a_{ij} is updated (to the new value $a_{ij}^{'}$) in the following way by gradient descent:

$$a_{ij}^{'} = a_{ij} + k\left(A_i x - y_i\right)x_j$$

where A_i is the *i*-th line of matrix *A* and *k* determines the descent step size (*k<0* and incorporates the factor of 2 resulting from the differentiation). The gradient descent needs to be run multiple times over all vector pairs *(x,y)* for the matrix to converge to an acceptable approximation which is called the transformation matrix *T*.

3.2 Synthetic Children's Model Creation

The transformation matrix can be applied to the female model in order to create a new synthetic acoustic model which should suit children's speech better than adult acoustic models. It is unlikely that the transformation applied "as is" will result in the best model possible, therefore the transformation can be altered (amplified or weakened) in order to yield the best results. An intuitive way to alter the impact of the transformation is taking the matrix *T* to a certain power *p*. Synthetic models can be created by applying T^p to the female model[1], for various values *p*. If children's data is available for evaluation purposes, the best value of *p* can be determined. The power *p* is claimed to be language independent. It might vary in nuances, but experi-

[1] Taking a matrix to the power of *p* is meant in the sense $T^{p^{1/p}} = T$, $T^0 = Identity$, $T^1 = T$

ments have shown that a value around 0.25 is a reasonable choice.

3.3 Transformation Algorithm

The previous section presented the theoretical means necessary for the synthetic children's model creation process. The precise, patent-pending algorithm to create a synthetic children's model in a new language is as follows (Hagen et al., 2008):

1. Train a male and a female acoustic model
2. Estimate the transform T from the male to the female model
3. Determine the power p by which the transform T should be adjusted
4. Apply T^p to the female acoustic model to create the synthetic children's model

Step 3, the determination of the power p, can be done in two different ways. If children's test data in the relevant language is available, various models based on different p-values can be evaluated and the best one chosen. If there is no children's data available in a new language, p can be estimated by evaluations in a language where there is enough male, female, and children's speech data available. The claim here is that the power p is relatively language independent and estimating p in a different language is superior to a simple guess.

4 Experiments

The algorithm was tested on two languages: US English and Spanish. For both languages sufficient male, female, and children's speech data was available (more than 20 hours) in order to train valid acoustic models and to have reference children's acoustic models available. For English test data we used a corpus of 22 native speakers in the age range of 5 to 14. The number of utterances is 2,182. For Spanish test data the corpus is comprised of 19 speakers in the age range of 8 to 13 years. The number of utterances is 2,598.

The transform from the male to the female model was estimated in English. The power of p was gradually increased and the transformation matrix was adjusted. With this adjusted matrix T^p a synthetic children's model was built. This synthetic children's model was evaluated on children's test data and the results were compared to the reference children's model's and the female model's performance.

When speech is evaluated in a language learning system, the first step is utterance verification, meaning the task of evaluating if the user actually tried to produce the desired utterance. The Equal Error Rate (EER) on the utterance level is a means of evaluating this performance. For each utterance an in- and out-of-grammar likelihood score is determined. The EER operating points, determined by the cutting point of the two distributions (in-grammar and out-of-grammar), are reported as an error metric. Figure 1 shows the EER values of the synthetic model applied to children's data.

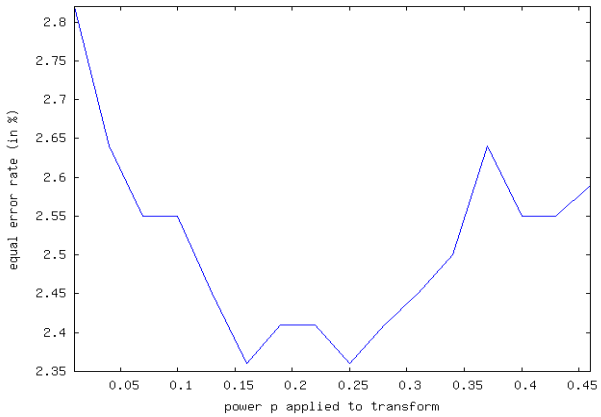


Figure 1: Synthetic model's EER performance depending on the power p used for model creation.

It can be seen that the best performance is reached at about $p=0.25$. The overview of the results is given in Table 1.

	Equal Error Rate
Real Children's Model	1.90%
Male Model	4.07%
Female Model	2.92%
Synthetic Model	2.36%

Table 1: EER numbers when using a real children's model compared to a male, female, and synthetic model for children's data evaluation.

The results show that the synthetic children's model yields good classification results when applied to children's data. The gold standard, the real children's model application, results in the best EER performance.

If the same evaluation scenario is applied to Spanish, a very similar picture evolves. Figure 2 shows the EER results versus transformation power p for Spanish children's data.

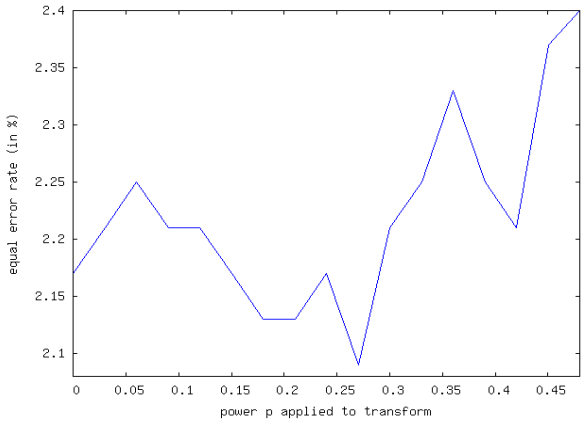


Figure 2: Spanish synthetic model's EER performance depending on the power p used for model creation.

In Figure 2 it can be seen that the optimal setting for *p* is about *0.27*. This value is very similar to the one found for US English, which supports, but certainly does not prove, the language independence claim. Results for Spanish are given in Table 2.

	Equal Error Rate
Real Children's model	2.40%
Male model	5.62%
Female model	2.17%
Synthetic model	2.09%

Table 2: EER numbers for Spanish when using a real children's model compared to a male, female, and synthetic model for Spanish children's data evaluation.

Similar to English, the Spanish synthetic model performs better than the female model on children's speech. Interestingly, the acoustic model purely trained on children's data performs worse than the female and the synthetic model. It is not clear why the children's model does not outperform the female and the synthetic model; an explanation could be diverse and variable training data that hurts classification performance.

It can be seen that for US English and Spanish the power *p* used to adjust the transformation is about *0.25*. Therefore, for a new language where only adult data is available, the transformation from the male to the female model can be estimated and applied to the female model (after being adjusted by *p=0.25*). The resulting synthetic model will work reasonably well and could be refined as soon as children's data becomes available.

5 Conclusion

This work presented a new technique to create children's acoustic models from adult acoustic models without the need for children's training data when applied to a new language. While it can be assumed that the availability of children's data would improve the resulting acoustic models, the approach is effective if children's data is not available. It will be interesting to see how performance of this technique compares to adapting adult models by adaptation techniques, i.e. MLLR, when limited amounts of children's data are available. Two scenarios are possible: With increasing amount of children's data speaker adaptation will draw even and/or be superior. The other possibility is that the presented technique yields better results regardless how much real children's data is available, due to the higher variability and noise-pollution of children's data.

References

Claes, T., Dologlou, I, ten Bosch, L., Van Compernolle, D. 1997. *New Transformations of Cepstral Parameters for Automatic Vocal Tract Length Normalization in Speech Recognition*, 5th Europ. Conf. on Speech Comm. and Technology, Vol. 3: 1363-1366.

Cosi, P., Pellom, B. 2005. *Italian children's speech recognition for advanced interactive literacy tutors*. Proceedings Interspeech, Lisbon, Portugal.

Elenius, D. and Blomberg, M. 2005. *Adaptation and Normalization Experiments in Speech Recognition for 4 to 8 Year old Children*. Proceedings Interspeech, Lisbon, Portugal.

Gustafson, J., Sjölander, K. 2002. *Voice transformations for improving children²s speech recognition in a publicly available dialogue system*. ICSLP, Denver.

Hagen, A., Pellom, B., and Cole, R. 2003. *Children's Speech Recognition with Application to Interactive Books and Tutors*. Proceedings ASRU, USA.

Lee, S., Potamianos, A., and Narayanan, S. 1999. *Acoustics of children's speech: Developmental changes of temporal and spectral parameter*. J. Acoust. Soc. Am., Vol. 105(3):1455-1468.

Pitz, M., Ney, H. 2005. *Vocal Tract Normalization Equals Linear Transformation in Cepstral Space.* IEEE Trans. Speech & Audio Proc., 13(5): 930-944.

Potamianos, A., Narayanan, S., and Lee, S. 1997. *Automatic Speech Recognition for Children*. Proceedings Eurospeech, Rhodes, Greece.

Hagen, A., Pellom, B., and Hacioglu, K. 2008. *Method for Creating a Speech Model*. US Patent Pending.

Detecting Pitch Accents at the Word, Syllable and Vowel Level

Andrew Rosenberg
Columbia University
Department of Computer Science
amaxwell@cs.columbia.edu

Julia Hirschberg
Columbia University
Department of Computer Science
julia@cs.columbia.edu

Abstract

The automatic identification of prosodic events such as *pitch accent* in English has long been a topic of interest to speech researchers, with applications to a variety of spoken language processing tasks. However, much remains to be understood about the best methods for obtaining high accuracy detection. We describe experiments examining the optimal **domain** for accent analysis. Specifically, we compare pitch accent identification at the syllable, vowel or word level as domains for analysis of acoustic indicators of accent. Our results indicate that a word-based approach is superior to syllable- or vowel-based detection, achieving an accuracy of 84.2%.

1 Introduction

Prosody in a language like Standard American English can be used by speakers to convey semantic, pragmatic and paralinguistic information. Words are made intonationall prominent, or *accented* to convey information such as contrast, focus, topic, and information status. The communicative implications of accenting influence the interpretation of a word or phrase. However, the acoustic excursions associated with accent are typically aligned with the lexically stressed syllable of the accented word. This disparity between the domains of acoustic properties and communicative impact has led to different approaches to pitch accent detection, and to the use of different domains of analysis.

In this paper, we compare automatic pitch accent detection at the vowel, syllable, and word level to determine which approach is optimal. While lexical and syntactic information has been shown to contribute to the detection of pitch accent, we only explore acoustic features. This decision allows us to closely examine the indicators of accent that are present in the speech signal in isolation from linguistic effects that may indicate that a word or syllable may be accented. The choice of domain for automatic pitch accent prediction it also related to how that prediction is to be used and impacts how it can be evaluated in comparison with other research efforts. While some downstream spoken language processing tasks benefit by knowing *which* syllable in a word is accented, such as clarification of communication misunderstandings, such as "I said **un**lock the door – not lock it!", most applications care only about which *word* is intonationally prominent. For the identification of contrast, given/new status, or focus, only word-level information is required. While the performance of nucleus- or syllable-based predictions can be translated to word predictions, such a translation is rarely performed, making it difficult to compare performance and thus determine which approach is best.

In this paper, we describe experiments in pitch accent detection comparing the use of vowel nuclei, syllables and words as units of analysis. In Section 2, we discuss related work. We describe the materials in Section 3, the experiments themselves in Section 4 and conclude in Section 5.

2 Related Work

Acoustic-based approaches to pitch accent detection have explored prediction at the word, syllable, and

Proceedings of NAACL HLT 2009: Short Papers, pages 81–84,
Boulder, Colorado, June 2009.

vowel level, but have rarely compared prediction accuracies across these different domains. An exception is the work of Ross and Ostendorf (1996), who detect accent on the Boston University Radio News Corpus (BURNC) at both the syllable and word level. Using CART predictions as input to an HMM, they detect pitch accents on syllables spoken by a single speaker from BURNC with 87.7% accuracy, corresponding to 82.5% word-based accuracy, using both lexical and acoustic features. In comparing the discriminative usefulness of syllables vs. syllable nuclei for accent detection, Tamburini (2003) finds syllable nuclei (vowel) duration to be as useful to full syllables. Rosenberg and Hirschberg (2007) used an energy-based ensemble technique to detect pitch accents with 84.1% accuracy on the read portion of the Boston Directions Corpus, without using lexical information. Sridhar *et al.* (2008) obtain 86.0% word-based accuracy using maximum entropy models from acoustic and syntactic information on the BURNC. Syllable-based detection by Ananthakrishnan and Narayanan (2008) combines acoustic, lexical and syntactic FSM models to achieve a detection rate of 86.75%. Similar suprasegmental features have also been explored in work at SRI/ICSI which employs a hidden event model to model intonational information for a variety of tasks including punctuation and disfluency detection (Baron et al., 2002). However, while progress has been made in accent detection performance in the past 15 years, with both word and syllable accuracy at about 86%, these accuracies have been achieved with different methods and some have included lexico-syntactic as well as acoustic features. It is still not clear which domain of acoustic analysis provides the most accurate cues for accent prediction. To address this issue, our work compares accent detection at the syllable nucleus, full syllable, and word levels, using a common modeling technique and a common corpus, to focus on the question of which domain of acoustic analysis is most useful for pitch accent prediction.

3 Boston University Radio News Corpus

Our experiments use 157.9 minutes (29,578 words) from six speakers in the BURNC (Ostendorf et al., 1995) recordings of professionally read radio news. This corpus has been prosodically annotated with full ToBI labeling (Silverman et al., 1992), including the presence and type of accents; these are annotated at the syllable level and 54.7% (16,178) of words are accented. Time-aligned phone boundaries generated by forced alignment are used to identify vowel regions for analysis. There are 48,359 vowels in the corpus and 34.8 of these are accented. To generate time-aligned syllable boundaries, we align the forced-aligned phones with a syllabified lexicon included with the corpus.

The use of BURNC for comparative accent prediction in our three domains is not straightforward, due to anomalies in the corpus. First, the lexicon and forced-alignment output in BURNC use distinct phonetic inventories; to align these, we have employed a *minimum edit distance* procedure where aligning any two vowels incurs zero cost. This guarantees that, at a minimum the vowels will be aligned correctly. Also, the number of syllables per word in the lexicon does not always match the number of vowels in the forced alignment. This leads to 114 syllables containing two forced-aligned vowels, and 8 containing none. Instead of performing *post hoc* correction of the syllabification results, we include all of the automatically identified syllables in the data set. This syllabification approach generates 48,253 syllables, 16,781 (34.8%) bearing accent.

4 Pitch Accent Detection Experiments

We train logistic regression models to detect the presence of pitch accent using acoustic features drawn from each word, syllable and vowel, using Weka (Witten et al., 1999). The features we use included pitch (f0), energy and duration, which have been shown to correlate with pitch accent in English. To model these, we calculate pitch and energy contours for each token using Praat (Boersma, 2001). Duration information is derived using the vowel, syllable or word segmentation described in Section 3. The feature vectors we construct include features derived from both raw and speaker z-score normalized[1] pitch and energy contours. The feature vector used in all three analysis scenarios is comprised of minimum, maximum, mean, standard de-

[1]Z-score normalization:$x_{norm} = \frac{x-\mu}{\sigma}$, where x is a value to normalize, μ and σ are mean and standard deviation. These are estimated from all pitch or intensity values for a speaker.

viation and the z-score of the maximum of these raw and normalized acoustic contours. The duration of the region in seconds is also included.

The results of ten-fold cross validation classification experiments are shown in Table 1. Note that, when running ten-fold cross validation on syllables and vowels, we divide the folds by words, so that each syllable within a word is a member of the same fold. To allow for direct comparison of the three approaches, we generate word-based results from vowel- and syllable-based experiments. If any syllable or vowel in a word is hypothesized as accented, the containing word is predicted to be accented. Vowel/syllable accuracies should be higher

Region	Accuracy (%)	F-Measure
Vowel	68.5 ± 0.319	0.651 ± 0.00329
Syllable	75.6 ± 0.125	0.756 ± 0.00188
Word	82.9 ± 0.168	0.845 ± 0.00162

Table 1: *Word-level accuracy and F-Measure*

than word-based accuracies since the baseline is significantly higher. However, we find that the F-measure for detecting accent is consistently higher for word-based results. A prediction of **accented** on any component syllable is sufficient to generate a correct word prediction.

Our results suggest, first of all, that there is discriminative information beyond the syllable nucleus. Syllable-based classification is significantly better than vowel-based classification, whether we compare accuracy or F-measure. It is possible that the narrow region of analysis offered by syllable and vowel-based analysis makes the aggregated features more susceptible to the effects of noise. Moreover, errors in the forced-alignment phone boundaries and syllabification may negatively impact the performance of vowel- and syllable-based approaches. Until automatic phone alignment improves, word-based prediction appears to be more reliable. An automatic, acoustic syllable-nucleus detection approach may be able generate more discriminative regions of analysis for pitch accent detection than the forced-alignment and lexicon alignment technique used here. This remains an area for future study.

However, if we accept that the feature representations accurately model the acoustic information contained in the regions of analysis and that the BURNC annotation is accurate, the most likely explanation for the superiority of word-based prediction over syllable- or vowel-based strategiesis is that the acoustic excursions correlated with accent occur outside a word's lexically stressed syllable. In particular, complex pitch accents in English are generally realized on multiple syllables. To examine this possibility, we looked at the distribution of misses from the three classification scenarios. The distribution of pitch accent types of missed detections using evaluation of the three scenarios is shown in Table 2. In the ToBI framework, the complex pitch accents include L+H*, L*+H, H+!H* and their downstepped variants. As we suspected, larger units of analysis lead to improved performance on complex tones; χ^2 analysis of the difference between the error distributions yields a χ^2 of 42.108, p< 0.0001.

Since accenting is the perception of a word as more prominent than surrounding words, features that incorporate local contextual acoustic information should improve detection accuracy at all levels. To represent surrounding acoustic context in feature vectors, we calculate the z-score of the maximum and mean pitch and energy over six regions. Three of these are "short range" regions: one previous region, one following region, and both the previous and following region. The other three are "long range" regions. For words, these regions are defined as two previous words, two following words, and both two previous and two following words. To give syllable- and vowel-based classification scenarios access to a comparable amount of acoustic context, the "long range" regions covered ranges of three syllables or vowels. There are approximately 1.63 syllables/vowels per word in the BURNC corpus; thus, on balance, a window of two words is equivalent to one of three syllables. Duration is also normalized relative to the duration of regions within the contextual regions. Accuracy and f-measure results from ten-fold cross validation experiments are shown in Table 3. We find dramatic

Analysis Region	Accuracy (%)	F-Measure
Vowel	77.4 ± 0.264	0.774 ± 0.00370
Syllable	81.9 ± 0.197	0.829 ± 0.00195
Word	84.2 ± 0.247	0.858 ± 0.00276

Table 3: *Word-level accuracy and F-Measure with Contextual Features*

increases in the performance of vowel- and syllable-

Region	H*	L*	Complex	Total Misses
Vowel	.6825 (3732)	.0686 (375)	.2489 (1361)	1.0 (5468)
Syllable	.7033 (2422)	.0851 (293)	.2117 (729)	1.0 (3444)
Word	.7422 (2002)	.0610 (165)	.1986 (537)	1.0 (2704)

Table 2: *Distribution of missed detections organized by H*, L* and complex pitch accents.*

based performance when we include contextual features. Vowel-based classification shows nearly 10% absolute increase accuracy when translated to the word level. The improvements in word-based classification, however, are less dramatic. It may be that word-based analysis already incorporates much the contextual information that is helpful for detecting pitch accents. The feature representations in each of these three experiments include a comparable amount of acoustic context. This suggests that the superiority of word-based detection is not simply due to the access to more contextual information, but rather that there is discriminative information outside the accent-bearing syllable.

5 Conclusion and Future Work

In this paper, we describe experiments comparing the detection of pitch accents on three acoustic domains – words, syllables and vowels – using acoustic features alone. To permit direct comparison between accent prediction in these three domains of analysis, we generate word-, syllable-, and vowel-based results directly, and then transfer syllable- and nucleus-based predictions to word predictions.

Our experiments show that word-based accent detection significantly outperforms syllable- and vowel-based approaches. Extracting features that incorporate acoustic information from surrounding context improves performance in all three domains. We find that there is, in fact, acoustic information discriminative to pitch accent that is found within accented words, outside the accent-bearing syllable. We achieve 84.2% word-based accuracy — significantly below the 86.0% reported by Sridhar *et al.* (2008) using syntactic and acoustic components. However, our experiments use only acoustic features, since we are concerned with comparing domains of acoustic analysis within the larger task of accent identification. Our 84.2% accuracy *is* significantly higher than the 80.09% accuracy obtained by the 10ms frame-based acoustic modeling described in (Sridhar et al., 2008). Our aggregations of pitch and energy contours over a region of analysis appear to be more helpful than short frame modeling.

In future work, we will explore a number of techniques to transfer word based predictions to syllables. This will allow us to compare word-based detection to published syllable-based results. Preliminary results suggest that word-based detection is superior regardless of the domain of evaluation.

References

S. Ananthakrishnan and S. Narayanan. 2008. Automatic prosodic event detection using acoustic, lexical and syntactic evidence. *IEEE Transactions on Audio, Speech & Language Processing*, 16(1):216–228.

D. Baron, E. Shriberg, and A. Stolcke. 2002. Automatic punctuation and disfluency detection in multi-party meetings using prosodic and lexical cues. In *ICSLP*.

P. Boersma. 2001. Praat, a system for doing phonetics by computer. *Glot International*, 5(9-10):341–345.

M. Ostendorf, P. Price, and S. Shattuck-Hufnagel. 1995. The boston university radio news corpus. Technical Report ECS-95-001, Boston University, March.

A. Rosenberg and J. Hirschberg. 2007. Detecting pitch accent using pitch-corrected energy-based predictors. In *Interspeech*.

K. Ross and M. Ostendorf. 1996. Prediction of abstract prosodic labels for speech synthesis. *Computer Speech & Language*, 10(3):155–185.

K. Silverman, M. Beckman, J. Pitrelli, M. Ostendorf, C. Wightman, P. Price, J. Pierrehumbert, and J. Hirschberg. 1992. Tobi: A standard for labeling english prosody. In *Proc. of the 1992 International Conference on Spoken Language Processing*, volume 2, pages 12–16.

V. R. Sridhar, S. Bangalore, and S. Narayanan. 2008. Exploiting acoustic and syntactic features for prosody labeling in a maximum entropy framework. *IEEE Transactions on Audio, Speech & Language Processing*, 16(4):797–811.

F. Tamburini. 2003. Prosodic prominence detection in speech. In *ISSPA2003*, pages 385–388.

I. Witten, E. Frank, L. Trigg, M. Hall, G. Holmes, and S. Cunningham. 1999. Weka: Practical machine learning tools and techniques with java implementation. In *ICONIP/ANZIIS/ANNES International Workshop*, pages 192–196.

Shallow Semantic Parsing for Spoken Language Understanding

Bonaventura Coppola and **Alessandro Moschitti** and **Giuseppe Riccardi**
Department of Information Engineering and Computer Science - University of Trento, Italy
{coppola,moschitti,riccardi}@disi.unitn.it

Abstract

Most Spoken Dialog Systems are based on speech grammars and frame/slot semantics. The semantic descriptions of input utterances are usually defined ad-hoc with no ability to generalize beyond the target application domain or to learn from annotated corpora. The approach we propose in this paper exploits machine learning of frame semantics, borrowing its theoretical model from computational linguistics. While traditional automatic Semantic Role Labeling approaches on written texts may not perform as well on spoken dialogs, we show successful experiments on such porting. Hence, we design and evaluate automatic FrameNet-based parsers both for English written texts and for Italian dialog utterances. The results show that disfluencies of dialog data do not severely hurt performance. Also, a small set of FrameNet-like manual annotations is enough for realizing accurate Semantic Role Labeling on the target domains of typical Dialog Systems.

1 Introduction

Commercial services based on spoken dialog systems have consistently increased both in number and in application scenarios (Gorin et al., 1997). Despite its success, current Spoken Language Understanding (SLU) technology is mainly based on simple conceptual annotation, where just very simple semantic composition is attempted. In contrast, the availability of richer semantic models as FrameNet (Baker et al., 1998) is very appealing for the design of better dialog managers. The first step to enable the exploitation of frame semantics is to show that accurate automatic semantic labelers can be designed for processing conversational speech.

In this paper, we face the problem of performing shallow semantic analysis of speech transcriptions from real-world dialogs. In particular, we apply Support Vector Machines (SVMs) and Kernel Methods to the design of a semantic role labeler (SRL) based on FrameNet. Exploiting Tree Kernels (Collins and Duffy, 2002; Moschitti et al., 2008), we can quickly port our system to different languages and domains. In the experiments, we compare results achieved on the English FrameNet against those achieved on a smaller Italian FrameNet-like corpus of spoken dialog transcriptions. They show that the system is robust enough to disfluencies and noise, and that it can be easily ported to new domains and languages.

In the remainder of the paper, Section 2 presents our basic Semantic Role Labeling approach, Section 3 describes the experiments on the English FrameNet and on our Italian dialog corpus, and Section 4 draws the conclusions.

2 FrameNet-based Semantic Role Labeling

Semantic frames represent prototypical events or situations which individually define their own set of actors, or frame participants. For example, the COMMERCE_SCENARIO frame includes participants as SELLER, BUYER, GOODS, and MONEY. The task of FrameNet-based shallow semantic parsing can be implemented as a combination of multiple specialized semantic labelers as those in (Carreras and Màrquez, 2005), one for each frame. Therefore, the general semantic parsing work-flow includes 4 main steps: (i) *Target Word Detection*, where the semantically relevant words bringing predicative information (the frame *targets*) are detected, e.g. the verb *to purchase* for the above example; (ii) *Frame Disambiguation*, where the correct frame for every target word (which may be ambiguous) is determined, e.g. COMMERCE_SCENARIO; (iii) *Boundary Detection (BD)*, where the sequences of words realizing the frame elements (or predicate

Proceedings of NAACL HLT 2009: Short Papers, pages 85–88,
Boulder, Colorado, June 2009.

arguments) are detected; and (iv) *Role Classification (RC)* (or argument classification), which assigns semantic labels to the frame elements detected in the previous step, e.g. GOODS. Therefore, we implement the full task of FrameNet-based parsing by a combination of multiple specialized SRL-like labelers, one for each frame (Coppola et al., 2008). For the design of each single labeler, we use the state-of-the-art strategy developed in (Pradhan et al., 2005; Moschitti et al., 2008).

2.1 Standard versus Structural Features

In machine learning tasks, the manual engineering of effective features is a complex and time consuming process. For this reason, our SVM-based SRL approach exploits the combination of two different models. We first used Polynomial Kernels over handcrafted, linguistically-motivated, *"standard"* SRL features (Gildea and Jurafsky, 2002; Pradhan et al., 2005; Xue and Palmer, 2004). Nonetheless, since we aim at modeling an SRL system for a new language (Italian) and a new domain (dialog transcriptions), the above features may result ineffective. Thus, to achieve independence on the application domain, we exploited Tree Kernels (Collins and Duffy, 2002) over automatic structural features proposed in (Moschitti et al., 2005; Moschitti et al., 2008). These are complementary to standard features and are obtained by applying Tree Kernels (Collins and Duffy, 2002; Moschitti et al., 2008) to basic tree structures expressing the syntactic relation between arguments and predicates.

3 Experiments

Our purpose is to show that an accurate automatic FrameNet parser can be designed with reasonable effort for Italian conversational speech. For this purpose, we designed and evaluated both a semantic parser for the English FrameNet (Section 3.1) and one for a corpus of Italian spoken dialogs (Section 3.2). The accuracy of the latter and its comparison against the former can provide evidence to sustain out thesis or not.

3.1 Evaluation on the English FrameNet

In this experiment we trained and tested boundary detectors (BD) and role classifiers (RC) as described in Section 2. More in detail, (a) we trained 5 BDs according to the syntactic categories of the possible target predicates, namely nouns, verbs, adjectives, adverbs and prepositions; (b) we trained 782 one-versus-all multi-role classifiers RC, one for each available frame and predicate syntactic category, for a total of 5,345 binary classifiers; and (c) we applied the above models for recognizing predicate arguments and their associated semantic labels in sentences, where the frame label and the target predicate were considered as given.

3.1.1 Data Set

We exploited the FrameNet 1.3 data base. After preprocessing and parsing the sentences with Charniak's parser, we obtained 135,293 semantically-annotated and syntactically-parsed sentences.

The above dataset was partitioned into three subsets: 2% of data (2,782 sentences) for training the BDs, 90% (121,798 sentences) for training RC, and 1% (1,345 sentences) as test set. The remaining data were discarded. Accordingly, the number of positive and negative training examples for BD were: 2,764 positive and 37,497 negative examples for verbal, 1,189 and 35,576 for nominal, 615 and 14,544 for adjectival, 0 and 40 for adverbial, and 7 and 177 for prepositional predicates (for a total of 4,575 and 87,834). For RC, the total numbers were 207,662 and 1,960,423, which divided by the number of role types show the average number of 39 positive versus 367 negative examples per role label.

3.1.2 Results

We tested several kernels over standard features (Gildea and Jurafsky, 2002; Pradhan et al., 2005) and structured features (Moschitti et al., 2008): the Polynomial Kernel (*PK*, with a degree of 3), the Tree Kernel (TK) and its combination with the bag of word kernel on the tree leaves (TKL). Also, the combinations PK+TK and PK+TKL were tested.

The 4 rows of Table 1 report the performance of different classification tasks. They show in turn: (1) the "pure" performance of the BD classifiers, i.e. considering correct the classification decisions also when a correctly classified tree *node* does not exactly correspond to its argument's *word* boundaries. Such mismatch frequently happens when the parse tree (which is automatically generated) contains in-

	PK			TK			PK+TK			TKL			PK+TKL		
Eval sett.	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
BD	.887	.675	.767	.949	.652	.773	.915	.698	**.792**	.938	.659	.774	.908	.701	.791
BD pj	.850	.647	.735	.919	.631	.748	.875	.668	**.758**	.906	.636	.747	.868	.670	.757
BD+RC	.654	.498	.565	.697	.479	.568	.680	.519	**.588**	.689	.484	.569	.675	.521	.588
BD+RC pj	.625	.476	.540	.672	.462	.548	.648	.495	**.561**	.663	.466	.547	.644	.497	.561

Table 1: Results on FrameNet dataset: Polynomial Kernel, two different Tree Kernels, and their combinations (see Section 3.1.2) with 2% training for BD and 90% for RC.

correct node attachments; (2) the real performance of the BD classification when actually "projected" ("pj") on the tree leaves, i.e. when matching not only the constituent node as in 1, but also exactly matching the selected *words* (leaves) with those in the FrameNet gold standard. This also implies the exact automatic syntactic analysis for the subtree; (3) the same as in (1), with the argument role classification (RC) also performed (frame element labels must also match); (4) the same as in (2), with RC also performed. For each classification task, the Precision, Recall and F_1 measure achieved by means of different kernel combinations are shown in the columns of the table. Only for the best configuration in Table 1 (PK+TK, results in bold) the amount of training data for the BD model was increased from 2% to 90%, resulting in a popular splitting for this task(Erk and Pado, 2006). Results are shown in Table 2: the PK+TK kernel achieves 1.0 Precision, 0.732 Recall, and 0.847 F_1. These figures can be compared to 0.855 Precision, 0.669 Recall and 0.751 F_1 of the system described in (Erk and Pado, 2006) and trained over the same amount of data. In conclusion, our best learning scheme is currently capable of tagging FrameNet data with exact boundaries and role labels at 63% F_1. Our next steps will be (1) further improving the RC models using FrameNet-specific information (such as Frame and role inheritance), and (2) introducing an effective Frame classifier to automatically choose Frame labels.

Enhanced PK+TK			
Eval Setting	P	R	F_1
BD (nodes)	1.0	.732	.847
BD (words)	.963	.702	.813
BD+RC (nodes)	.784	.571	.661
BD+RC (words)	.747	.545	.630

Table 2: Results on the FrameNet dataset. Best configuration from Table 1, raised to 90% of training data for BD and RC.

Eval Setting	P	R	F_1	P	R	F_1
				PK		
BD	-	-	-	.900	.869	.884
BD+RC	-	-	-	.769	.742	.756
	TK			**PK+TK**		
BD	.887	.856	.871	.905	.873	**.889**
BD+RC	.765	.738	.751	.774	.747	**.760**

Table 3: Experiment Results on the Italian dialog corpus for different learning schemes and kernel combinations.

3.2 Evaluation on Italian Spoken Dialogs

In this section, we present the results of BD and RC of our FrameNet parser on the smaller Italian spoken dialog corpus. We assume here as well that the target word (i.e. the predicate for which arguments have to be extracted) along with the correct frame are given.

3.2.1 Data Set

The Italian dialog corpus includes 50 real human-human dialogs recorded and manually transcribed at the call center of the help-desk facility of an Italian Consortium for Information Systems. The dialogs are fluent and spontaneous conversations between a caller and an operator, concerning hardware and software problems. The dialog turns contain 1,677 annotated frame instances spanning 154 FrameNet frames and 20 new ad hoc frames specific for the domain. New frames mostly concern data processing such as NAVIGATION, DISPLAY_DATA, LOSE_DATA, CREATE_DATA. Being intended as a reference resource, this dataset includes partially human-validated syntactic analysis, i.e. lower branches corrected to fit arguments. We divided such dataset into 90% training (1,521 frame instances) and 10% testing (156 frame instances). Each frame instance brings its own set of frame participant (or predicate argument) instances.

For BD, the very same approach as in Section 3.1 was followed. For RC, we also followed the same approach but, in order to cope with data sparse-

ness, we also attempted a different RC strategy by merging data related to different syntactic predicates within the same frame. So, within each frame, we merged data related to verbal predicates, nominal predicates, and so on. Due to the short space available, we will just report results for this latter approach, which performed sensitively better.

3.2.2 Results

The results are reported in Table 3. Each table block shows Precision, Recall and F_1 for either PK, TK, or PK+TK. The rows marked as BD show the results for the task of marking the exact constituent boundaries of every frame element (argument) found. The rows marked as BD+RC show the results for the two-stage pipeline of *both* marking the exact constituent boundaries and *also* assigning the correct semantic label. A few observations hold.

First, the highest F_1 has been achieved using the PK+TK combination. On this concern, we underline that kernel combinations *always* gave the best performance in any experiment we run.

Second, we emphasize that the F_1 of PK is surprisingly high, since it exploits the set of standard SRL feature (Gildea and Jurafsky, 2002; Pradhan et al., 2005), originally developed for English and left unmodified for Italian. Nonetheless, their performance is comparable to the Tree Kernels and, as we said, their combination improves the result. Concerning the structured features exploited by Tree Kernels, we note that they work as well without any tuning when ported to Italian dialogs.

Finally, the achieved F_1 is extremely good. In fact, our corresponding result on the FrameNet corpus (Table 2) is P=0.784, R=0.571, F_1=0.661, where the corpus contains much more data, its sentences come from a standard written text (no disfluencies are present) and it is in English language, which is morphologically simpler than Italian. On the other hand, the Italian corpus includes optimal syntactic annotation which exactly fits the frame semantics, and the number of frames is lower than in the FrameNet experiment.

4 Conclusions

The good performance achieved for Italian dialogs shows that FrameNet-based parsing is viable for labeling conversational speech in any language using a few training data. Moreover, the approach works well for very specific domains, like helpdesk/customer conversations. Nonetheless, additional tests based on fully automatic transcription and syntactic parsing are needed. However, our current results show that future research on complex spoken dialog systems is enabled to exploit automatically generated frame semantics, which is our very direction.

Acknowledgments

The authors wish to thank Daniele Pighin for the SRL subsystem and Sara Tonelli for the Italian corpus. This work has been partially funded by the European Commission - LUNA project (contract n.33549), and by the Marie Curie Excellence Grant for the ADAMACH project (contract n.022593).

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of COLING-ACL '98*, pages 86–90.

Xavier Carreras and Lluís Màrquez. 2005. Introduction to the CoNLL-2005 Shared Task: Semantic Role Labeling. In *Proceedings of CoNLL-2005*.

Michael Collins and Nigel Duffy. 2002. New Ranking Algorithms for Parsing and Tagging: Kernels over Discrete structures, and the voted perceptron. In *ACL02*.

Bonaventura Coppola, Alessandro Moschitti, and Daniele Pighin. 2008. Generalized framework for syntax-based relation mining. In *IEEE-ICDM 2008*.

Katrin Erk and Sebastian Pado. 2006. Shalmaneser - a flexible toolbox for semantic role assignment. In *Proceedings of LREC 2006*, Genoa, Italy.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic Labeling of Semantic Roles. *Computational Linguistics*.

A. L. Gorin, G. Riccardi, and J. H. Wright. 1997. How may i help you? *Speech Communication*.

Alessandro Moschitti, Bonaventura Coppola, Daniele Pighin, and Roberto Basili. 2005. Engineering of syntactic features for shallow semantic parsing. In *ACL WS on Feature Engineering for ML in NLP*.

Alessandro Moschitti, Daniele Pighin, and Roberto Basili. 2008. Tree kernels for semantic role labeling. *Computational Linguistics*, 34(2):193–224.

Sameer Pradhan, Kadri Hacioglu, Valerie Krugler, Wayne Ward, James H. Martin, and Daniel Jurafsky. 2005. Support Vector Learning for Semantic Argument Classification. *Machine Learning*.

Nianwen Xue and Martha Palmer. 2004. Calibrating features for semantic role labeling. In *Proceedings of EMNLP 2004*.

Automatic Agenda Graph Construction from Human-Human Dialogs using Clustering Method

Cheongjae Lee, Sangkeun Jung, Kyungduk Kim, Gary Geunbae Lee
Department of Computer Science and Engineering
Pohang University of Science and Technology
Pohang, South Korea
{lcj80,hugman,getta,gblee}@postech.ac.kr

Abstract

Various knowledge sources are used for spoken dialog systems such as task model, domain model, and agenda. An agenda graph is one of the knowledge sources for a dialog management to reflect a discourse structure. This paper proposes a clustering and linking method to automatically construct an agenda graph from human-human dialogs. Preliminary evaluation shows our approach would be helpful to reduce human efforts in designing prior knowledge.

1 Introduction

Data-driven approaches have been long applied for spoken language technologies. Although a data-driven approach requires time-consuming data annotation, the training is done automatically and requires little human supervision. These advantages have motivated the development of data-driven dialog modelings (Williams and Young, 2007, Lee et al., 2009). In general, the data-driven approaches are more robust and portable than traditional knowledge-based approaches. However, various knowledge sources are still used in many spoken dialog systems that have been developed recently. These knowledge sources contain task model, domain model, and agenda which are powerful representation to reflect the hierarchy of natural dialog control. In the spoken dialog systems, these are manually designed for various purposes including dialog modeling (Bohus and Rudnicky, 2003, Lee et al., 2008), search space reduction (Young et al., 2007), domain knowledge (Roy and Subramaniam, 2006), and user simulation (Schatzmann et al., 2007).

We have proposed an example-based dialog modeling (EBDM) framework using an agenda graph as prior knowledge (Lee et al., 2008). This is one of the data-driven dialog modeling techniques and the next system action is determined by selecting the most similar dialog examples in dialog example database. In the EBDM framework for task-oriented dialogs, agenda graph is manually designed to address two aspects of a dialog management: (1) Keeping track of the dialog state with a view to ensuring steady progress towards task completion, and (2) Supporting n-best recognition hypotheses to improve the robustness of dialog manager. However, manually building such graphs for various applications may be labor intensive and time consuming. Thus, we have tried to investigate how to build this graph automatically. Consequently, we sought to solve the problem by automatically building the agenda graph using clustering method from an annotated dialog corpus.

2 Related Work

Clustering techniques have been widely used to build prior knowledge for spoken dialog systems. One of them is automatic construction of domain model (or topic structure) which is one of the important resources to handle user's queries in call centers. Traditional approach to building domain models is that the analysts manually generate a domain model through inspection of the call records. However, it has recently been proposed to use an unsupervised technique to generate domain models automatically from call transcriptions (Roy and Subramaniam, 2006). In addition, there has been research on how to automatically learn models of task-oriented discourse structure using dialog act and task information (Bangalore et al., 2006). Discourse structure is necessary for dialog state-specific speech recognition and language understanding to improve the performance by predicting the next possible dialog states. In addition, the discourse structure is essential to determine whether the current utterance in the dialog is part of the current subtask or starts a new task.

Proceedings of NAACL HLT 2009: Short Papers, pages 89–92,
Boulder, Colorado, June 2009.

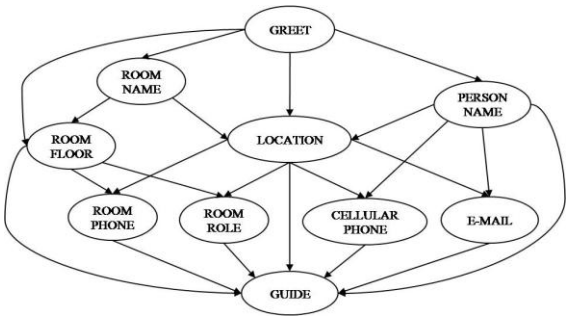


Figure 1: Example of an agenda graph for building guidance domain

Feature Types	Features	#Size
Word-level features	unigram	175
	bigram	573
	trigram	1034
Utterance-level features	dialog act (DA)	9
	main goal (MG)	16
	slot filling status	8
	system act (SA)	26
Discourse-level features	previous DA	10
	previous MG	17
	previous SA	27

Table 1: List of feature sets

More recently, it has been proposed stochastic dialog management such as the framework of a partially observable Markov decision process (POMDP). This framework is statistically data-driven and theoretically principled dialog modeling. However, detailed dialog states in the master space should be clustered into general dialog states in summary space to scale up POMDP-based dialog management for practical applications (Williams and Young, 2007). To address this problem, an unsupervised automatic clustering of dialog states has been introduced and investigated in POMDP-based dialog manager (Lefevre and Mori, 2007).

In this paper, we are also interested in exploring methods that would automatically construct the agenda graph as prior knowledge for the EBDM framework.

3 Agenda Graph

In this section, we begin with a brief overview of EBDM framework and agenda graph. The basic idea of the EBDM is that the next system action is predicted by finding semantically similar user utterance in the dialog state space. The agenda graph was adapted to take into account the robustness problem for practical applications. Agenda graph *G* is a simply a way of encoding the domain-specific dialog control to complete the task. *G* is represented by a directed acyclic graph (DAG) (Figure 1). An agenda is one of the subtask flows, which is a possible path from root node to terminal node. *G* is composed of nodes (*v*) which correspond to possible intermediate steps in the process of completing the specified task, and edges (*e*) which connect nodes. In other words, v corresponds to dialog state to achieve domain-specific subtask in its expected agenda. Each node includes three different components: (1) A precondition that must be true before the subtask is executed; (2) A description of the node that includes its label and identifier; and (3) Links to nodes that will be executed at the subsequent turn. In this system, this graph is used to rescore n-best ASR hypotheses and to interpret the discourse state such as new task, next task, and new subtask based on topological position on the graph. In the agenda graph *G*, each node holds a set of relevant dialog examples which may appear in the corresponding dialog states when a precondition of the node is true. To determine the next system action, the dialog manager first generates possible candidate nodes with n-best hypotheses by using a discourse interpretation algorithm based on the agenda graph, and then selects the focus node which is the most likely dialog state given the previous dialog state. Finally the best example in the focus node is selected to determine appropriate system action.

Human efforts are required to manually design the agenda graph to integrate it into the EBDM framework. However, it is difficult to define all possible precondition rules and to assign the transition probabilities to each link based only on the discretion of the system developer. To solve these problems, we tried to construct the agenda graph from the annotated dialog corpus using clustering technique.

4 Clustering and Linking

4.1 Node Clustering

Each precondition has been manually defined to map relevant dialog examples into each node. To avoid this, the dialog examples are automatically grouped into the closest cluster (or node) by a node clustering. In this section, we explain a feature extraction and clustering method for constructing the agenda graph.

4.1.1 Feature Extraction

Each dialog example should be converted into a feature vector for a node clustering. To represent the feature vectors, we first extract all n-grams which occur more frequently than a threshold and do not contain any stop word as word-level features. We also extract utterance-level and discourse-level features from the annotated dialog corpus to reflect semantic and contextual information because a dialog state can be characterized using semantic and contextual information derivable from the annotations. The utterance is thus characterized by the set of various features as shown in Table 1.

For a set of N dialog examples $X=\{x_i|i=1,..,N\}$, the binary feature vectors are represented by using a set of features from the dialog corpus. To calculate the distance of two feature vectors, we used a cosine measure as a binary vector distance measure:

$$d(x_i, x_j) = 1 - \frac{(x_i \cdot x_j)}{\|x_i\| \cdot \|x_j\|}$$

where x_i and x_j denoted two feature vectors. However, each feature vector contains small number of non-zero terms (<20 features) compared to the feature space (>2000 features). Therefore, most pairs of utterances share no common feature, and their distance is close to 1.0. To address this sparseness problem, the distance between two utterances can be computed by checking only the non-zero terms of corresponding feature vectors (Liu, 2005).

4.1.2 Clustering

After extracting feature vectors from the dialog corpus, we used K-means clustering algorithm which is the simplest and most commonly used algorithm employing a squared error criterion. At the initialization step, one cluster mean is randomly selected in the data set and k-1 means are iteratively assigned by selecting the farthest point from pre-selected centers as the following equation:

$$u_k = \arg\max_{x \in X} \left(\sum_{i=1}^{k-1} d(x, u_i) \right)$$

where each cluster c_k is represented as a mean vector u_k. At the assignment step, each example is assigned to the nearest cluster $\hat{c}_t$ by minimizing the distance of cluster mean u_k and dialog example x_t.

$$\hat{c}_t = \arg\min_{1 \le k \le K} (d(u_k, x_t))$$

The responsibilities r_{kt} of each cluster c_k are calculated for each example x_t as the following rule:

$$r_{kt} = \frac{\exp\{-\beta \cdot d(u_k, x_t)\}}{\sum_l \exp\{-\beta \cdot d(u_l, x_t)\}}$$

where β is the stiffness and usually assigned to 1.

During the update step, the means are recomputed using the current cluster membership by reflecting their responsibilities:

$$u_k = \frac{\sum_t r_{kt} x_t}{\sum_t r_{kt}}$$

4.2 Node Linking

From the node clustering step, node v_k for cluster c_k is obtained from the dialog corpus and each node contains similar dialog examples by the node clustering algorithm. Next, at the node linking step, each node should be connected with an appropriate transition probability to build the agenda graph which is a DAG (Figure 2).

Algorithm : *NodeLinking*(*C*)

Input : C = a set of clusters $\{c_k \mid c_k \in C,\ k=1, ..., K\}$

Output : $G(V, E)$ = agenda graph (directed acyclic graph)

where each $v_k \in V$ is subtask node, each $e_{ij} \in E$ is its directed edges

1 $V \leftarrow$ GENERATENODES(C)
2 $V \leftarrow V \cup \{v_{root}, v_{terminal}\}$
3 $E \leftarrow$ INTRODUCEDGES(V,C)
4 **for** $i \leftarrow 1$ **to** K
5 **do for** $j \leftarrow 1$ **to** K
6 **do** $f(v_i, v_j) = n(x \in v_i \rightarrow v_j) / n(x \in v_i)$
7 **while** all edges are visited
8 **do** $e_{ij} \leftarrow$ FINDMINIMALEDGE(E)
9 **if** $f(v_i, v_j) < \delta$ **then** $E \leftarrow E - \{e_{ij}\}$ *(underweighted edge pruning)*
10 **else if** ISCONNECTED(v_j, v_i) **then** $E \leftarrow E - \{e_{ij}\}$ *(cycle deletion)*
11 **for** $i \leftarrow 1$ **to** K
12 **do for** $j \leftarrow 1$ **to** K
13 **do** $p(v_j \mid v_i) = f(v_i, v_j) / \sum_t f(v_i, v_t)$
14 **return** $G = (V, E)$

Figure 2: Node Linking Algorithm

This linking information can come from the dialog corpus because the task-oriented dialogs consist of sequential utterances to complete the tasks. Using sequences of dialog examples obtained with the dialog corpus, relative frequencies of all outgoing edges are calculated to weight directed edges:

$$f(v_i, v_j) = \frac{n(x \in v_i \rightarrow v_j)}{n(x \in v_i)}$$

where $n(x \in v_i)$ represents the number of dialog examples in v_i and $n(x \in v_i \rightarrow v_j)$ denotes the number of dialog examples having directed edge from v_i to v_j. Next some edges are pruned when the weight falls below a pre-defined threshold δ, and the cycle paths are removed by deleting minimal edge in cycle paths through a depth-first traversal. Finally the transition probability can be estimated by normalizing relative frequencies with the remained edges.

$$p(v_j \mid v_i) = \frac{f(v_i, v_j)}{\sum_l f(v_i, v_l)}$$

5 Experiment & Result

A spoken dialog system for intelligent robot was developed to provide information about building (e.g., room number, room name, room type) and people (e.g., name, phone number, e-mail address). If the user selects a specific room to visit, then the robot takes the user to the desired room. For this system, we collect a human-human dialog corpus of about 880 user utterances from 214 dialogs which were based on a set of pre-defined 10 subjects relating to building guidance task. Then, we designed an agenda graph and integrated it into the EBDM framework. In addition, a simulated environment with a user simulator and an ASR channel (Jung et

al., 2008) was developed to evaluate our approach by simulating a realistic scenario.

First we measured the clustering performance to verify our approach for constructing the agenda graph. We used the manually clustered examples by a set of precondition rules as the reference clusters. Table 2 shows error rates when different feature sets are used for *K*-means clustering in which *K* is equal to 10 because a hand-crafted graph included 10 nodes. The error rate was significantly reduced when using all feature sets.

Feature sets	Error rate (%)
Word-level features	46.51
+Utterance-level features	34.63
+Discourse-level features	31.20

Table 2: Error rates for node clustering (*K*=10)

We also evaluated the dialog system performance with the agenda graphs which are manually (HC-AG) or automatically designed (AC-AG). We also used 10-best recognition hypotheses with 20% word error rate (WER) for a dialog management and 1000 simulated dialogs for an automatic evaluation. In this result, although the system with HC-AG slightly outperforms the system with AC-AG, we believe that AC-AG can be helpful to manage task-oriented dialogs with less human costs for designing the hand-crafted agenda graph.

System	TCR (%)	AvgUserTurn
Using HC-AG	92.96	4.41
Using AC-AG	89.95	4.39

Table 3: Task completion rate (TCR) and average user turn (AvgUserTurn) (WER=20%)

6 Conclusion & Discussion

In this paper, we address the problem of automatic knowledge acquisition of agenda graph to structure task-oriented dialogs. We view this problem as a first step in clustering the dialog states, and then in linking between each cluster based on the dialog corpus. The experiment results show that our approach can be applicable to easily build the agenda graph for prior knowledge.

There are several possible subjects for further research on our approach. We can improve the clustering performance by using a distance metric learning algorithm to consider the correlation between features. We can also discover hidden links in the graph by exploring new dialog flows with random walks.

Acknowledgement

This research was supported by the MKE (Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute for Information Technology Advancement) (IITA-2009-C1090-0902-0045).

References

Bangalore, S., Fabbrizio, G.D. and Stent, A. 2006. Learning the structure of task-driven human-human dialogs. *Proc. of the Association for Computational Linguistics*, 201-208.

Bohus, B. and Rudnicky, A. 2003. RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda. *Proc. of the European Conference on Speech, Communication and Technology*, 597-600.

Jung, S., Lee, C., Kim, K. and Lee, G.G. 2008. An Integrated Dialog Simulation Technique for Evaluating Spoken Dialog Systems. *Proc. of Workshop on Speech Processing for Safety Critical Translation and Pervasive Applications, International Conference on Computational Linguistics*, 9-16.

Lee, C., Jung, S. and Lee, G.G. 2008. Robust Dialog Management with N-best Hypotheses using Dialog Examples and Agenda. *Proc. of the Association for Computational Linguistics*, 630-637.

Lee, C., Jung, S., Kim, S. and Lee, G.G. 2009. Example-based Dialog Modeling for Practical Multi-domain Dialog System. *Speech Communication*, 51(5):466-484.

Lefevre, F. and Mori, R.D. 2007. Unsupervised State Clustering for Stochastic Dialog Management. *Proc. of the IEEE Workshop on Automatic Speech Recognition and Understanding*, 550-555.

Liu, Z. 2005. An Efficient Algorithm for Clustering Short Spoken Utterances. *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 593-596.

Roy, S. and Subramaniam, L.V. 2006. Automatic generation of domain models for call centers from noisy transcriptions. *Proc. of the Association for Computational Linguistics*, 737-744.

Schatzmann, J., Thomson, B., Weilhammer, K., Ye, H. and Young, S. 2007. Agenda-based User Simulation for Bootstrapping a POMDP Dialogue System. *Proc. of the Human Language Technology/North American Chapter of the Association for Computational Linguistics*, 149-152.

Williams, J.D. and Young, S. 2007. Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21:393-422.

Young, S., Schatzmann, J., Weilhammer, K. and Ye, H. 2007. The Hidden Information State Approach to Dialog Management. *Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing*, 149-152.

A Simple Sentence-Level Extraction Algorithm for Comparable Data

Christoph Tillmann and Jian-ming Xu
IBM T.J. Watson Research Center
Yorktown Heights, N.Y. 10598
{ctill,jianxu}@us.ibm.com

Abstract

The paper presents a novel sentence pair extraction algorithm for comparable data, where a large set of candidate sentence pairs is scored directly at the sentence-level. The sentence-level extraction relies on a very efficient implementation of a simple symmetric scoring function: a computation speed-up by a factor of 30 is reported. On Spanish-English data, the extraction algorithm finds the highest scoring sentence pairs from close to 1 trillion candidate pairs without search errors. Significant improvements in BLEU are reported by including the extracted sentence pairs into the training of a phrase-based SMT (Statistical Machine Translation) system.

1 Introduction

The paper presents a simple sentence-level translation pair extraction algorithm from comparable monolingual news data. It differs from similar algorithms that select translation correspondences explicitly at the document level (Fung and Cheung, 2004; Resnik and Smith, 2003; Snover et al., 2008; Munteanu and Marcu, 2005; Quirk et al., 2007; Utiyama and Isahara, 2003). In these publications, the authors use Information-Retrieval (IR) techniques to match document pairs that are likely translations of each other. More complex sentence-level models are then used to extract parallel sentence pairs (or fragments). From a computational perspective, the document-level filtering steps are needed to reduce the number of candidate sentence pairs. While IR techniques might be useful to improve the selection accuracy, the current paper demonstrates that they are not necessary to obtain parallel sentence pairs. For some data, e.g. the Portuguese-English Reuters data used in the experiments in Section 3, document-level information may not even be available.

In this paper, sentence pairs are extracted by a simple model that is based on the so-called IBM Model-1 (Brown et al., 1993). The Model-1 is trained on some parallel data available for a language pair, i.e. the data used to train the baseline systems in Section 3. The scoring function used in this paper is inspired by phrase-based SMT. Typically, a phrase-based SMT system includes a feature that scores phrase pairs using lexical weights (Koehn et al., 2003) which are computed for two directions: source to target and target to source. Here, a sentence pair is scored as a phrase pair that covers all the source and target words. The scoring function $\varrho(S,T)$ is defined as follows:

$$\begin{aligned}\varrho(S,T) &= \qquad (1)\\ &= \sum_{j=1}^{J} \underbrace{\frac{1}{J} \cdot \; log(\; \overbrace{\frac{1}{I} \cdot \sum_{i=1}^{I} p(s_j|t_i)}^{p(s_j|T)} \;)}_{\sigma(s_j,T)} \; + \\ &\quad \sum_{i=1}^{I} \underbrace{\frac{1}{I} \cdot \; log(\; \overbrace{\frac{1}{J} \cdot \sum_{j=1}^{J} p(t_i|s_j)}^{p(t_i|S)} \;)}_{\tau(t_i,S)}\end{aligned}$$

Proceedings of NAACL HLT 2009: Short Papers, pages 93–96,
Boulder, Colorado, June 2009.

Here, $S = s_1^J$ is the source sentence of length J and $T = t_1^I$ is the target sentence of length I. $p(s|T)$ is the Model-1 probability assigned to the source word s given the target sentence T, $p(t|S)$ is defined accordingly. $p(s|t)$ and $p(t|s)$ are word translation probabilities obtained by two parallel Model-1 training steps on the same data, but swapping the role of source and target language. They are smoothed to avoid 0.0 entries; there is no special NULL-word model and stop words are kept. The $\log(\cdot)$ is applied to turn the sentence-level probabilities into scores. These log-probabilities are normalized with respect to the source and target sentence length: this way the score $\varrho(S,T)$ can be used across all sentence pairs considered, and a single manually set threshold θ is used to select all those sentence pairs whose score is above it. For computational reasons, the sum $\varrho(S,T)$ is computed over the following terms: $\tau(t_i,S)$ where $1 \leq i \leq I$ and $\sigma(s_j,T)$, where $1 \leq j \leq J$. The τ's and σ's represent partial score contributions for a given source or target position. Note that $\varrho(S,T) \leq 0$ since the terms $\tau(\cdot,S) \leq 0$ and $\sigma(\cdot,T) \leq 0$.

Section 2 presents an efficient implementation of the scoring function in Eq. 1. Its effectiveness is demonstrated in Section 3. Finally, Section 4 discusses future work and extensions of the current algorithm.

2 Sentence-Level Processing

We process the comparable data at the sentence-level: for each language and all the documents in the comparable data, we distribute sentences over a list of files : one file for each news feed f (for the Spanish Gigaword data, there are 3 news feeds) and publication date d . The Gigaword data comes annotated with sentence-level boundaries, and all document boundaries are discarded. This way, the Spanish data consists of about 24 thousand files and the English data consists of about 53 thousand files (for details, see Table 2). For a given source sentence S, the search algorithm computes the highest scoring sentence pair $\varrho(S,T)$ over a set of candidate translations $T \in \Theta$, where $|\Theta|$ can be in the hundreds of thousands of sentences . Θ consists of all target sentences that have been published from the same news feed f within a 7 day window from the publication date of the current source sentence S. The extraction algorithm is guaranteed to find the highest scoring sentence pairs (S,T) among all $T \in \Theta$. In order to make this processing pipeline feasible, the scoring function in Eq. 1 needs to be computed very efficiently. That efficiency is based on the decomposition of the scoring functions into $I + J$ terms (τ's and σ's) where source and target terms are treated differently. While the scoring function computation is symmetric, the processing is organized according the source language files: all the source sentences are processed one-by-one with respect to their individual candidate sets Θ:

- **Caching for target term $\tau(t,S)$:** For each target word t that occurs in a candidate translation T, the Model-1 based probability $p(t|S)$ can be *cached*: its value is independent of the other words in T. The same word t in different target sentences is processed with respect to the same source sentence S and $p(t|S)$ has to be computed only once.

- **Array access for source terms $\sigma(s,T)$:** For a given source sentence S, we compute the scoring function $\varrho(S,T)$ over a set of target sentences $T \in \Theta$. The computation of the source term $\sigma(s,T)$ is based on translation probabilities $p(s|t)$. For each source word s, we can retrieve all target words t for which $p(s|t) > 0$ just **once**. We store those words t along with their probabilities in an array the size of the target vocabulary. Words t that do not have an entry in the lexicon have a 0 entry in that array. We keep a separate array for each source position. This way, we reduce the probability access to a simple array look-up. Generating the full array presentation requires less than 50 milliseconds per source sentence on average.

- **Early-Stopping:** Two loops compute the scoring function $\varrho(S,T)$ exhaustively for each sentence pair (S,T): 1) a loop over all the target position terms $\tau(t_i,S)$, and 2) a loop over all source position terms $\sigma(s_j,T)$. Once the current partial sum is lower than the best score $\varrho(S,T_{best})$ computed so far, the computation can be safely discarded as $\tau(t_i,S), \sigma(s_j,T) \leq$

Table 1: Effect of the implementation techniques on a full search that computes $\varrho(S,T)$ exhaustively for all sentence pairs (S,T) for a given S.

Implementation Technique	Speed [secs/sent]
Baseline	33.95
+ Array access source terms	19.66
+ Cache for target terms	3.83
+ Early stopping	1.53
+ Frequency sorting	1.23

Table 2: Corpus statistics for comparable data. Any document-level information is ignored.

	Spanish	English
Date-Feed Files	24,005	53,373
Sentences	19.4 million	47.9 million
Words	601.5 million	1.36 billion
	Portuguese	**English**
Date-Feed Files	351	355
Sentences	366.0 thousand	5.3 million
Words	11.6 million	171.1 million

0 and adding additional terms can only lower that partial sum further.

- **Frequency-Sorting:** Here, we aim at making the early pruning step more efficient. Source and target words are sorted according to the source and target vocabulary frequency: less frequent words occur at the beginning of a sentence. These words are likely to contribute terms with high partial scores. As a result, the early-stopping step fires earlier and becomes more effective.

- **Sentence-level filter:** The word-overlap filter in (Munteanu and Marcu, 2005) has been implemented: for a sentence pair (S,T) to be considered parallel the ratio of the lengths of the two sentences has to be smaller than two. Additionally, at least half of the words in each sentence have to have a translation in the other sentence based on the word-based lexicon. Here, the implementation of the coverage restriction is tightly integrated into the above implementation: the decision whether a target word is covered can be cached. Likewise, source word coverage can be decided by a simple array look-up.

3 Experiments

The parallel sentence extraction algorithm presented in this paper is tested in detail on the large-scale Spanish-English Gigaword data (Graff, 2006; Graff, 2007). The Spanish data comes from 3 news feeds: Agence France-Presse (AFP), Associated Press Worldstream (APW), and Xinhua News Agency (XIN). We do not use the additional news feed present in the English data. Table 1 demonstrates the effectiveness of the implementation techniques in Section 2. Here, the average extraction time per source sentence is reported for one of the $24,000$ source language files. This file contains 913 sentences. Here, the size of the target candidate set Θ is $61\,736$ sentences. All the techniques presented result in some improvement. The baseline uses only the length-based filtering and the coverage filtering without caching the coverage decisions (Munteanu and Marcu, 2005). Caching the target word probabilities results in the biggest reduction. The results are representative: finding the highest scoring target sentence T for a given source sentence S takes about 1 second on average. Since 20 million source sentences are processed, and the workload is distributed over roughly 120 processors, overall processing time sums to less than 3 days. Here, the total number of translation pairs considered is close to 1 trillion.

The effect of including additional sentence pairs along with selection statistics is presented in Table 3. Translation results are presented for a standard phrase-based SMT system. Here, both languages use a test set with a single reference. Including about 1.4 million sentence pairs extracted from the Gigaword data, we obtain a statistically significant improvement from 42.3 to 45.6 in BLEU (Papineni et al., 2002). The baseline system has been trained on about 1.8 million sentence pairs from Europarl and FBIS parallel data. We also present results for a Portuguese-English system: the baseline has been trained on Europarl and JRC data. Parallel sentence pairs are extracted from comparable Reuters news data published in 2006. The corpus statistics for

Table 3: Spanish-English and Portuguese-English extraction results.

Data Source	# candidates	#train pairs	Bleu
Spanish-English: $\theta = -4.1$			
Baseline	-	$1,825,709$	42.3
+ Gigaword	$955.5 \cdot 10^9$	$1,372,124$	45.6
Portuguese-English: $\theta = -5.0$			
Baseline	-	$2,221,891$	45.3
+ Reuters 06	$32.8 \cdot 10^9$	$48,500$	48.5

the Portuguese-English data are given in Table 2. The selection threshold θ is determined with the help of bilingual annotators (it typically takes a few hours). Sentence pairs are selected with a conservative threshold θ' first. Then, all the sentence pairs are sorted by descending score. The annotator descends this list to determine a score threshold cut-off. Here, translation pairs are considered to be parallel if 75 % of source and target words have a corresponding translation in the other sentence. Using a threshold $\theta = -4.1$ for the Spanish-English data, results in a selection precision of around 80 % (most of the misqualified pairs are partial translations with less than 75 % coverage or short sequences of high frequency words). This simple selection criterion proved sufficient to obtain the results presented in this paper. As can be seen from Table 3, the optimal threshold is language specific.

4 Future Work and Discussion

In this paper, we have presented a novel sentence-level pair extraction algorithm for comparable data. We use a simple symmetrized scoring function based on the Model-1 translation probability. With the help of an efficient implementation, it avoids any translation candidate selection at the document level (Resnik and Smith, 2003; Smith, 2002; Snover et al., 2008; Utiyama and Isahara, 2003; Munteanu and Marcu, 2005; Fung and Cheung, 2004). In particular, the extraction algorithm works when no document-level information is available. Its usefulness for extracting parallel sentences is demonstrated on news data for two language pairs. Currently, we are working on a feature-rich approach (Munteanu and Marcu, 2005) to improve the sentence-pair selection accuracy. Feature functions will be 'light-weight' such that they can be computed efficiently in an incremental way at the sentence-level. This way, we will be able to maintain our search-driven extraction approach. We are also re-implementing IR-based techniques to preselect translation pairs at the document-level, to gauge the effect of this additional filtering step. We hope that a purely sentence-level processing might result in a more productive pair extraction in future.

References

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The Mathematics of Statistical Machine Translation: Parameter Estimation. *CL*, 19(2):263–311.

Pascale Fung and Percy Cheung. 2004. Mining Very-Non-Parallel Corpora: Parallel Sentence and Lexicon Extraction via Bootstrapping and EM. In *Proc, of EMNLP 2004*, pages 57–63, Barcelona, Spain, July.

Dave Graff. 2006. *LDC2006T12: Spanish Gigaword Corpus First Edition.* LDC.

Dave Graff. 2007. *LDC2007T07: English Gigaword Corpus Third Edition.* LDC.

Philipp Koehn, Franz J. Och, and Daniel Marcu. 2003. Statistical Phrase-Based Translation. In *Proc. of HLT-NAACL'03*, pages 127–133, Edmonton, Alberta, Canada, May 27 - June 1.

Dragos S. Munteanu and Daniel Marcu. 2005. Improving Machine Translation Performance by Exploiting Non-Parallel Corpora. *CL*, 31(4):477–504.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *In Proc. of ACL'02*, pages 311–318, Philadelphia, PA, July.

Chris Quirk, Raghavendra Udupa, and Arul Menezes. 2007. Generative Models of Noisy Translations with Applications to Parallel Fragment Extraction. In *Proc. of the MT Summit XI*, pages 321–327, Copenhagen,Demark, September.

Philip Resnik and Noah Smith. 2003. The Web as Parallel Corpus. *CL*, 29(3):349–380.

Noah A. Smith. 2002. From Words to Corpora: Recognizing Translation. In *Proc. of EMNLP02*, pages 95–102, Philadelphia, July.

Matthew Snover, Bonnie Dorr, and Richard Schwartz. 2008. Language and Translation Model Adaptation using Comparable Corpora. In *Proc. of EMNLP08*, pages 856–865, Honolulu, Hawaii, October.

Masao Utiyama and Hitoshi Isahara. 2003. Reliable Measures for Aligning Japanese-English News Articles and Sentences. In *Proc. of ACL03*, pages 72–79, Sapporo, Japan, July.

Learning Combination Features with L_1 Regularization

Daisuke Okanohara† **Jun'ichi Tsujii**†‡§
†Department of Computer Science, University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo, Japan
‡School of Informatics, University of Manchester
§NaCTeM (National Center for Text Mining)
{hillbig,tsujii}@is.s.u-tokyo.ac.jp

Abstract

When linear classifiers cannot successfully classify data, we often add *combination features*, which are products of several original features. The searching for effective combination features, namely *feature engineering*, requires domain-specific knowledge and hard work. We present herein an efficient algorithm for learning an L_1 regularized logistic regression model with combination features. We propose to use the grafting algorithm with efficient computation of gradients. This enables us to find optimal weights efficiently without enumerating all combination features. By using L_1 regularization, the result we obtain is very compact and achieves very efficient inference. In experiments with NLP tasks, we show that the proposed method can extract effective combination features, and achieve high performance with very few features.

1 Introduction

A linear classifier is a fundamental tool for many NLP applications, including logistic regression models (LR), in that its score is based on a linear combination of features and their weights,. Although a linear classifier is very simple, it can achieve high performance on many NLP tasks, partly because many problems are described with very high-dimensional data, and high dimensional weight vectors are effective in discriminating among examples.

However, when an original problem cannot be handled linearly, *combination features* are often added to the feature set, where combination features are products of several original features. Examples of combination features are, word pairs in document classification, or part-of-speech pairs of head and modifier words in a dependency analysis task. However, the task of determining effective combination features, namely *feature engineering*, requires domain-specific knowledge and hard work.

Such a non-linear phenomenon can be implicitly captured by using the kernel trick. However, its computational cost is very high, not only during training but also at inference time. Moreover, the model is not interpretable, in that effective features are not represented explicitly. Many kernels methods assume an L_2 regularizer, in that many features are equally relevant to the tasks (Ng, 2004).

There have been several studies to find efficient ways to obtain (combination) features. In the context of boosting, Kudo (2004) have proposed a method to extract complex features that is similar to the item set mining algorithm. In the context of L_1 regularization. Dudík (2007), Gao (2006), and Tsuda (2007) have also proposed methods by which effective features are extracted from huge sets of feature candidates. However, their methods are still very computationally expensive, and we cannot directly apply this kind of method to a large-scale NLP problem.

In the present paper, we propose a novel algorithm for learning of an L_1 regularized LR with combination features. In our algorithm, we can exclusively extract effective combination features without enumerating all of the candidate features. Our method relies on a grafting algorithm (Perkins and Theeiler, 2003), which incrementally adds features like boosting, but it can converge to the global optimum.

We use L_1 regularization because we can obtain a sparse parameter vector, for which many of the parameter values are exactly zero. In other words, learning with L_1 regularization naturally has an intrinsic effect of feature selection, which results in an

Proceedings of NAACL HLT 2009: Short Papers, pages 97–100,
Boulder, Colorado, June 2009.

efficient and interpretable inference with almost the same performance as L_2 regularization (Gao et al., 2007).

The heart of our algorithm is a way to find a feature that has the largest gradient value of likelihood from among the huge set of candidates. To solve this problem, we propose an example-wise algorithm with filtering. This algorithm is very simple and easy to implement, but effective in practice.

We applied the proposed methods to NLP tasks, and found that our methods can achieve the same high performance as kernel methods, whereas the number of active combination features is relatively small, such as several thousands.

2 Preliminaries

2.1 Logistic Regression Model

In this paper, we consider a multi-class logistic regression model (LR). For an input x, and an output label $y \in \mathcal{Y}$, we define a feature vector $\phi(x, y) \in R^m$.

Then in LR, the probability for a label y, given an input x, is defined as follows:

$$p(y|x; \boldsymbol{w}) = \frac{1}{Z(x, \boldsymbol{w})} \exp\left(\boldsymbol{w}^T \phi(x, y)\right), \quad (1)$$

where $\boldsymbol{w} \in R^m$ is a weight vector[1] corresponding to each input dimension, and $Z(x, \boldsymbol{w}) = \sum_y \exp(\boldsymbol{w}^T \phi(x, y))$ is the partition function.

We estimate the parameter $\boldsymbol{w}$ by a maximum likelihood estimation (MLE) with L_1 regularization using training examples $\{(x_1, y_1), \ldots, (x_n, y_n)\}$:

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w}} -L(\boldsymbol{w}) + C \sum_i |w_i| \quad (2)$$

$$L(\boldsymbol{w}) = \sum_{i=1\ldots n} \log p(y_i|x_i; \boldsymbol{w})$$

where $C > 0$ is the trade-off parameter between the likelihood term and the regularization term. This estimation is a convex optimization problem.

2.2 Grafting

To maximize the effect of L_1 regularization, we use the grafting algorithm (Perkins and Theeiler, 2003); namely, we begin with the empty feature set, and incrementally add effective features to the current problem. Note that although this is similar to the boosting algorithm for learning, the obtained result is always optimal. We explain the grafting algorithm here again for the sake of clarity.

The grafting algorithm is summarized in Algorithm 1.

In this algorithm we retain two variables; $\boldsymbol{w}$ stores the current weight vector, and H stores the set of features with a non-zero weight. Initially, we set $\boldsymbol{w} = \mathbf{0}$, and $H = \{\}$. At each iteration, the feature is selected that has the largest absolute value of the gradient of the likelihood. Let $v_k = \frac{\partial L(\boldsymbol{w})}{\partial w_k}$ be the gradient value of the likelihood of a feature k. By following the definition, the value v_k can be calculated as follows,

$$v_k = \sum_{i,y} \alpha_{i,y} \phi_k(x_i, y), \quad (3)$$

where $\alpha_{i,y} = I(y_i = y) - p(y_i|x_i; \boldsymbol{w})$ and $I(a)$ is 1 if a is true and 0 otherwise.

Then, we add $k^* = \arg\max_k |v_k|$ to H and optimize (2) with regard to H only. The solution $\boldsymbol{w}$ that is obtained is used in the next search. The iteration is continued until $|v_k^*| < C$.

We briefly explain why we can find the optimal weight by this algorithm. Suppose that we optimize (2) with all features, and initialize the weights using the results obtained from the grafting algorithm. Since all gradients of likelihoods satisfy $|v_k| \leq C$, and the regularization term pushes the weight toward 0 by C, any changes of the weight vector cannot increase the objective value in (2). Since (2) is the convex optimization problem, the local optimum is always the global optimum, and therefore this is the global optimum for (2)

The point is that, given an efficient method to estimate v_k^* without the enumeration of all features, we can solve the optimization in time proportional to the active feature, regardless of the number of candidate features. We will discuss this in the next section.

3 Extraction of Combination Features

This section presents an algorithm to compute, for combination features, the feature v_k^* that has the largest absolute value of the gradient.

We propose an element-wise extraction method, where we make use of the sparseness of the training data.

In this paper, we assume that the values of the combination features are less than or equal to the original ones. This assumption is typical; for example, it is made in the case where we use binary values for original and combination features.

[1] A bias term b is often considered by adding an additional dimension to $\phi(x, y)$

Algorithm 1 Grafting

Input: training data (x_i, y_i) $(i = 1, \cdots, n)$ and parameter C
$H = \{\}, \boldsymbol{w} = \mathbf{0}$
loop
  $\boldsymbol{v} = \frac{\partial L(\boldsymbol{w})}{\partial \boldsymbol{w}}$ ($L(\boldsymbol{w})$ is the log likelihood term)
  $k^* = \arg\max_k |v_k|$ (The result of Algorithm 2)
  if $|\boldsymbol{v}_{k^*}| < C$ **then** break
  $H = H \cup k^*$
  Optimize $\boldsymbol{w}$ with regards to H
end loop
Output $\boldsymbol{w}$ and H

First, we sort the examples in the order of their $\sum_y |\alpha_{i,y}|$ values. Then, we look at the examples one by one. Let us assume that r examples have been examined so far. Let us define

$$\boldsymbol{t} = \sum_{i \le r, y} \alpha_{i,y} \phi(x_i, y) \quad (4)$$

$$\boldsymbol{t}^- = \sum_{i > r, y} \alpha^-_{i,y} \phi(x_i, y) \quad \boldsymbol{t}^+ = \sum_{i > r, y} \alpha^+_{i,y} \phi(x_i, y)$$

where $\alpha^-_{i,y} = \min(\alpha_{i,y}, 0)$ and $\alpha^+_{i,y} = \max(\alpha_{i,y}, 0)$.

Then, simple calculus shows that the gradient value for a combination feature k, v_k, for which the original features are $k1$ and $k2$, is bounded below/above thus;

$$\begin{aligned} t_k + t^-_k < \quad & v_k \quad < t_k + t^+_k \quad (5) \\ t_k + \max(t^-_{k1}, t^-_{k2}) < \quad & v_k \quad < t_k + \min(t^+_{k1}, t^+_{k2}). \end{aligned}$$

Intuitively, the upper bound of (5) is the case where the combination feature fires only for the examples with $\alpha_{i,y} \ge 0$, and the lower bound of (5is the case where the combination feature fires only for the examples with $\alpha_{i,y} \le 0$. The second inequality arises from the fact that the value of a combination feature is equal to or less than the values of its original features. Therefore, we examine (5) and check whether or not $|v_k|$ will be larger than C. If not, we can remove the feature safely.

Since the examples are sorted in the order of their $\sum_y |\alpha_{i,y}|$, the bound will become tighter quickly. Therefore, many combination features are filtered out in the early steps. In experiments, the weights for the original features are optimized first, and then the weights for combination features are optimized. This significantly reduces the number of candidates for combination features.

Algorithm 2 Algorithm to return the feature that has the largest gradient value.

Input: training data (x_i, y_i) and its $\alpha_{i,y}$ value $(i = 1, \ldots, n,\ y = 1, \ldots, |\mathscr{Y}|)$, and the parameter C. Examples are sorted with respect to their $\sum_y |\alpha_{i,y}|$ values.
$\boldsymbol{t}^+ = \sum_{i=1}^n \sum_y \max(\alpha_{i,y}, 0)\phi(x, y)$
$\boldsymbol{t}^- = \sum_{i=1}^n \sum_y \min(\alpha_{i,y}, 0)\phi(x, y)$
$\boldsymbol{t} = \mathbf{0}$, $H = \{\}$ // Active Combination Feature
for $i = 1$ to n and $y \in \mathscr{Y}$ **do**
  for all combination features k in x_i **do**
    if $|v_k| > C$ (Check by using Eq.(5)) **then**
      $v_k := v_k + \alpha_{i,y}\phi_k(x_i, y)$
      $H = H \cup k$
    end if
  end for
  $\boldsymbol{t}^+ := \boldsymbol{t}^+ - \max(\alpha_{i,y}, 0)\phi(x_i, y)$
  $\boldsymbol{t}^- := \boldsymbol{t}^- - \min(\alpha_{i,y}, 0)\phi(x_i, y)$
end for
Output: $\arg\max_{k \in H} v_k$

Algorithm 2 presents the details of the overall algorithm for the extraction of effective combination features. Note that many candidate features will be removed just before adding.

4 Experiments

To measure the effectiveness of the proposed method (called L_1-Comb), we conducted experiments on the dependency analysis task, and the document classification task. In all experiments, the parameter C was tuned using the development data set.

In the first experiment, we performed Japanese dependency analysis. We used the Kyoto Text Corpus (Version 3.0), Jan. 1, 3-8 as the training data, Jan. 10 as the development data, and Jan. 9 as the test data so that the result could be compared to those from previous studies (Sassano, 2004)[2]. We used the shift-reduce dependency algorithm (Sassano, 2004). The number of training events was $11,3332$, each of which consisted of two word positions as inputs, and $y = \{0, 1\}$ as an output indicating the dependency relation. For the training data, the number of original features was 78570, and the number of combination features of degrees 2 and 3 was 5787361, and 169430335, respectively. Note that we need not see all of them using our algorithm.

[2]The data set is different from that in the CoNLL shared task. This data set is more difficult.

Table 1: The performance of the Japanese dependency task on the Test set. The active features column shows the number of nonzero weight features.

	DEP. ACC. (%)	TRAIN TIME (S)	ACTIVE FEAT.
L_1-COMB	89.03	605	78002
L_1-ORIG	88.50	35	29166
SVM 3-POLY	88.72	35720	(KERNEL)
L_2-COMB3	89.52	22197	91477782
AVE. PERCE.	87.23	5	45089

In all experiments, combination features of degrees 2 and 3 (the products of two or three original features) were used.

We compared our methods using LR with L_1 regularization using original features (L_1-Original), SVM with a 3rd-polynomial Kernel, LR with L_2 regularization using combination features with up to 3 combinations (L_2-Comb3), and an averaged perceptron with original features (Ave. Perceptron).

Table 1 shows the result of the Japanese dependency task. The accuracy result indicates that the accuracy was improved with automatically extracted combination features. In the column of active features, the number of active features is listed. This indicates that L_1 regularization automatically selects very few effective features. Note that, in training, L_1-Comb used around 100 MB, while L_2-Comb3 used more than 30 GB. The most time consuming part for L_1-Comb was the optimization of the L_1-LR problem.

Examples of extracted combination features include POS pairs of head and modifiers, such as *Head/Noun*-Modifier/Noun, and combinations of distance features with the POS of head.

For the second experiment, we performed the document classification task using the Tech-TC-300 data set (Davidov et al., 2004)[3]. We used the tf-idf scores as feature values. We did not filter out any words beforehand. The Tech-TC-300 data set consists of 295 binary classification tasks. We divided each document set into a training and a test set. The ratio of the test set to the training set was $1:4$. The average number of features for tasks was $25,389$.

Table 2 shows the results for L_1-LR with combination features and SVM with linear kernel[4]. The results indicate that the combination features are effective.

[3]http://techtc.cs.technion.ac.il/techtc300/techtc300.html

[4]SVM with polynomial kernel did not achieve significant improvement

Table 2: Document classification results for the Tech-TC-300 data set. The column F_2 shows the average of F_2 scores for each method of classification.

	F_2
L_1-COMB	0.949
L_1-ORIG	0.917
SVM (LINEAR KERNEL)	0.896

5 Conclusion

We have presented a method to extract effective combination features for the L_1 regularized logistic regression model. We have shown that a simple filtering technique is effective for enumerating effective combination features in the grafting algorithm, even for large-scale problems. Experimental results show that a L_1 regularized logistic regression model with combination features can achieve comparable or better results than those from other methods, and its result is very compact and easy to interpret. We plan to extend our method to include more complex features, and apply it to structured output learning.

References

Davidov, D., E. Gabrilovich, and S. Markovitch. 2004. Parameterized generation of labeled datasets for text categorization based on a hierarchical directory. In *Proc. of SIGIR*.

Dudík, Miroslav, Steven J. Phillips, and Robert E. Schapire. 2007. Maximum entropy density estimation with generalized regularization and an application to species distribution modeling. *JMLR*, 8:1217–1260.

Gao, J., H. Suzuki, and B. Yu. 2006. Approximation lasso methods for language modeling. In *Proc. of ACL/COLING*.

Gao, J., G. Andrew, M. Johnson, and K. Toutanova. 2007. A comparative study of parameter estimation methods for statistical natural language processing. In *Proc. of ACL*, pages 824–831.

Kudo, T. and Y. Matsumoto. 2004. A boosting algorithm for classification of semi-structured text. In *Proc. of EMNLP*.

Ng, A. 2004. Feature selection, l1 vs. l2 regularization, and rotational invariance. In *NIPS*.

Perkins, S. and J. Theeiler. 2003. Online feature selection using grafting. *ICML*.

Saigo, H., T. Uno, and K. Tsuda. 2007. Mining complex genotypic features for predicting HIV-1 drug resistance. *Bioinformatics*, 23:2455–2462.

Sassano, Manabu. 2004. Linear-time dependency analysis for japanese. In *Proc. of COLING*.

Multi-scale Personalization for Voice Search Applications

Daniel Bolaños
Center for Spoken Language Research
University of Colorado at Boulder, USA
bolanos@cslr.colorado.edu

Geoffrey Zweig
Microsoft Research
One Microsoft Way, Redmond, WA 98052
gzweig@microsoft.com

Patrick Nguyen
Microsoft Research
One Microsoft Way, Redmond, WA 98052
panguyen@microsoft.com

Abstract

Voice Search applications provide a very convenient and direct access to a broad variety of services and information. However, due to the vast amount of information available and the open nature of the spoken queries, these applications still suffer from recognition errors. This paper explores the utilization of personalization features for the post-processing of recognition results in the form of n-best lists. Personalization is carried out from three different angles: short-term, long-term and Web-based, and a large variety of features are proposed for use in a log-linear classification framework.

Experimental results on data obtained from a commercially deployed Voice Search system show that the combination of the proposed features leads to a substantial sentence error rate reduction. In addition, it is shown that personalization features which are very different in nature can successfully complement each other.

1 Introduction

Search engines are a powerful mechanism to find specific content through the use of queries. In recent years, due to the vast amount of information available, there has been significant research on the use of recommender algorithms to select what information will be presented to the user. These systems try to predict what content a user may want based not only on the user's query but on the user's past queries, history of clicked results, and preferences. In (Teevan et al., 1996) it was observed that a significant percent of the queries made by a user in a search engine are associated to a repeated search. Recommender systems like (Das et al., 2007) and (Dou et al., 2007) take advantage of this fact to refine the search results and improve the search experience.

In this paper, we explore the use of personalization in the context of voice searches rather than web queries. Specifically, we focus on data from a multimodal cellphone-based business search application (Acero et al., 2008). In such an application, repeated queries can be a powerful tool for personalization. These can be classified into short and long-term repetitions. Short-term repetitions are typically caused by a speech recognition error, which produces an incorrect search result and makes the user repeat or reformulate the query. On the other hand, long-term repetitions, as in text-based search applications, occur when the user needs to access some information that was accessed previously, for example, the exact location of a pet clinic.

This paper proposes several different user personalization methods for increasing the recognition accuracy in Voice Search applications. The proposed personalization methods are based on extracting short-term, long-term and Web-based features from the user's history. In recent years, other user personalization methods like deriving personalized pronunciations have proven successful in the context of mobile applications (Deligne et al., 2002).

The rest of this paper is organized as follows: Section 2 describes the classification method used for rescoring the recognition hypotheses. Section 3 describes the proposed personalization methods. Section 4 describes the experiments carried out. Finally,

Proceedings of NAACL HLT 2009: Short Papers, pages 101–104,
Boulder, Colorado, June 2009.

conclusions from this work are drawn in section 5.

2 Rescoring procedure

2.1 Log linear classification

Our work will proceed by using a log-linear classifier similar to the maximum entropy approach of (Berger and Della Pietra, 1996) to predict which word sequence W appearing on an n-best list N is most likely to be correct. This is estimated as

$$P(W|N) = \frac{\exp(\sum_i \lambda_i f_i(W,N))}{\sum_{W' \in N} \exp(\sum_i \lambda_i f_i(W',N))}. \quad (1)$$

The feature functions $f_i(W,N)$ can represent arbitrary attributes of W and N. This can be seen to be the same as a maximum entropy formulation where the class is defined as the word sequence (thus allowing potentially infinite values) but with sums restricted as a computational convenience to only those class values (word strings) appearing on the n-best list. The models were estimated with a widely available toolkit (Mahajan, 2007).

2.2 Feature extraction

Given the use of a log-linear classifier, the crux of our work lies in the specific features used. As a baseline, we take the hypothesis rank, which results in the 1-best accuracy of the decoder. Additional features were obtained from the personalization methods described in the following section.

3 Personalization methods

3.1 Short-term personalization

Short-term personalization aims at modeling the repair/repetition behavior of the user. Short-term features are a mechanism suitable for representing negative evidence: if the user repeats a utterance it normally means that the hypotheses in the previous n-best lists are not correct. For this reason, if a hypothesis is contained in a preceding n-best list, that hypothesis should be weighted negatively during the rescoring.

A straightforward method for identifying likely repetitions consists of using a fixed size time window and considering all the user queries within that window as part of the same repetition round. Once an appropriate window size has been determined, the proposed short-term features can be extracted for each hypothesis using a binary tree like the one depicted in figure 1, where feature values are in the leaves of the tree.

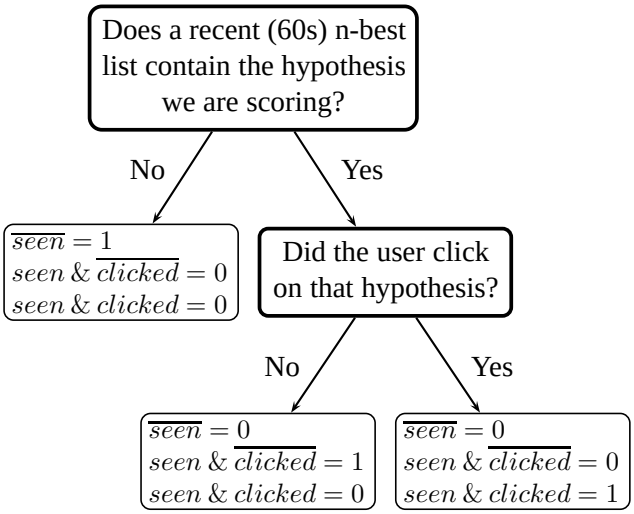


Figure 1: Short-term feature extraction (note that overlines mean "do not").

Given these features, we expect "seen and not clicked" to have a negative weight while "seen and clicked" should have a positive weight.

3.2 Long-term personalization

Long-term personalization consists of using the user history (i.e. recognition hypotheses that were confirmed by the user in the past) to predict which recognition results are more likely. The assumption here is that recognition hypotheses in the n-best list that match or "resemble" those in the user history are more likely to be correct. The following list enumerates the long-term features proposed in this work:

- User history (occurrences): number of times the hypothesis appears in the user history.
- User history (alone): 1 if the hypothesis appears in the user history and no other competing hypothesis does, otherwise 0.
- User history (most clicked): 1 if the hypothesis appears in the user history and was clicked more times than any other competing hypothesis.
- User history (most recent): 1 if the hypothesis appears in the user history and was clicked

more recently than any other competing hypothesis.

- User history (edit distance): minimum edit distance between the hypothesis and the closest query in the user history, normalized by the number of words.
- User history (words in common): maximum number of words in common between the hypothesis and each of the queries in the user history, normalized by the number of words in the hypothesis.
- User history (plural/singular): 1 if either the plural or singular version of the hypothesis appears in the user history, otherwise 0.
- Global history: 1 if the hypothesis has ever been clicked by any user, otherwise 0.
- Global history (alone): 1 if the hypothesis is the only one in the n-best that has ever been clicked by any user, otherwise 0.

Note that the last two features proposed make use of the "global history" which comprises all the queries made by any user.

3.3 LiveSearch-based features

Typically, users ask for businesses that exist, and if a business exists it probably appears in a Web document indexed by Live Search (Live Search, 2006). It is reasonable to assume that the relevance of a given business is connected to the number of times it appears in the indexed Web documents, and in this section we derive such features.

For the scoring process, an application has been built that makes automated queries to Live Search, and for each hypothesis in the n-best list obtains the number of Web documents in which it appears. Denoting by x the number of Web documents in which the hypothesis (the exact sequence of words, e.g. "tandoor indian restaurant") appears, the following features are proposed:

- Logarithm of the absolute count: $log(x)$.
- Search results rank: sort the hypotheses in the n-best list by their relative value of x and use the rank as a feature.
- Relative relevance (I): 1 if the hypothesis was not found and there is another hypothesis in the n-best list that was found more than 100 times, otherwise 0.
- Relative relevance (II): 1 if the the hypothesis appears fewer than 10 times and there is another hypothesis in the n-best list that appears more than 100 times, otherwise 0.

4 Experiments

4.1 Data

The data used for the experiments comprises 22473 orthographically transcribed business utterances extracted from a commercially deployed large vocabulary directory assistance system.

For each of the transcribed utterances two n-best lists were produced, one from the commercially deployed system and other from an enhanced decoder with a lower sentence error rate (SER). In the experiments, due to their lower oracle error rate, n-bests from the enhanced decoder were used for doing the rescoring. However, these n-bests do not correspond to the listings shown in the user's device screen (i.e. do not match the user interaction) so are not suitable for identifying repetitions. For this reason, the short term features were computed by comparing a hypothesis from the enhanced decoder with the original n-best list from the immediate past. Note that all other features were computed solely with reference to the n-bests from the enhanced decoder.

A rescoring subset was made from the original dataset using only those utterances in which the n-best lists contain the correct hypothesis (in any position) and have more than one hypothesis. For all other utterances, rescoring cannot have any effect. The size of the rescoring subset is 43.86% the size of the original dataset for a total of 9858 utterances. These utterances were chronologically partitioned into a training set containing two thirds and a test set with the rest.

4.2 Results

The baseline system for the evaluation of the proposed features consist of a ME classifier trained on only one feature, the hypothesis rank. The resulting sentence error rate (SER) of this classifier is that of the best single path, and it is 24.73%. To evaluate

the contribution of each of the features proposed in section 3, a different ME classifier was trained using that feature in addition to the baseline feature. Finally, another ME classifier was trained on all the features together.

Table 1 summarizes the Sentence Error Rate (SER) for each of the proposed features in isolation and all together respect to the baseline. "UH" stands for user history.

Features	**SER**
Hypothesis rank (baseline)	24.73%
base + repet. ($\overline{seen}$)	24.48%
base + repet. ($seen$ & $clicked$)	24.32%
base + repet. ($seen$ & $\overline{clicked}$)	24.73%
base + UH (occurrences)	**23.76%**
base + UH (alone)	23.79%
base + UH (most clicked)	23.73%
base + UH (most recent)	23.88%
base + UH (edit distance)	23.76%
base + UH (words in common)	24.60%
base + UH (plural/singular)	24.76%
base + GH	24.63%
base + GH (alone)	24.66%
base + Live Search (absolute count)	24.35%
base + Live Search (rank)	24.85%
base + Live Search (relative I)	**23.51%**
base + Live Search (relative II)	23.69%
base + all	**21.54%**

Table 1: Sentence Error Rate (SER) for each of the features in isolation and for the combination of all of them.

5 Conclusions

The proposed features reduce the SER of the baseline system by 3.19% absolute on the rescoring set, and by 1.40% absolute on the whole set of transcribed utterances.

Repetition based features are moderately useful; by incorporating them into the rescoring it is possible to reduce the SER from 24.73% to 24.32%. Although repetitions cover a large percentage of the data, it is believed that inconsistencies in the user interaction (the right listing is displayed but not confirmed by the user) prevented further improvement.

As expected, long-term personalization based features contribute to improve the classification accuracy. The UH (occurrences) feature by itself is able to reduce the SER in about a 1%.

Live Search has shown a very good potential for feature extraction. In this respect it is interesting to note that a right design of the features seems critical to take full advantage of it. The relative number of counts of one hypothesis respect to other hypotheses in the n-best list is more informative than an absolute or ranked count. A simple feature using this kind of information, like Live Search (relative I), can reduce the SER in more than 1% respect to the baseline.

Finally, it has been shown that personalization based features can complement each other very well.

References

Alex Acero, Neal Bernstein, Rob Chambers, Yun-Cheng Ju, Xiao Li, Julian Odell, Patrick Nguyen, Oliver Scholtz and Geoffrey Zweig. 2008. *Live Search for Mobile: Web Services by Voice on the Cellphone.* ICASSP 2008, March 31 2008-April 4 2008. Las Vegas, NV, USA.

Adam L. Berger; Vincent J. Della Pietra; Stephen A. Della Pietra. 1996. *A Maximum Entropy Approach to Natural Language Processing.* Computational Linguistics, 1996. 22(1): p. 39-72.

Abhinandan Das, Mayur Datar and Ashutosh Garg. 2007. *Google News Personalization: Scalable Online Collaborative Filtering.* WWW 2007 / Track: Industrial Practice and Experience May 8-12, 2007. Banff, Alberta, Canada.

Sabine Deligne, Satya Dharanipragada, Ramesh Gopinath, Benoit Maison, Peder Olsen and Harry Printz. 2002. *A robust high accuracy speech recognition system for mobile applications.* Speech and Audio Processing, IEEE Transactions on, Nov 2002, Volume: 10, Issue: 8, On page(s): 551- 561.

Zhicheng Dou, Ruihua Song, and Ji-Rong Wen. 2007. *A large-scale evaluation and analysis of personalized search strategies.* In WWW '07: Proceedings of the 16th international conference on World Wide Web, pages 581 - 590, New York, NY, USA, 2007. ACM Press.

Live Search. "http://www.live.com,".

Milind Mahajan. 2007. *Conditional Maximum-Entropy Training Tool* http://research.microsoft.com/en-us/downloads/9f199826-49d5-48b6-ba1b-f623ecf36432/.

Jaime Teevan, Eytan Adar, Rosie Jones and Michael A. S. Potts. 2007. *Information Re-Retrieval: Repeat Queries in Yahoos Logs.* SIGIR, 2007.

The Importance of Sub-Utterance Prosody in Predicting Level of Certainty

Heather Pon-Barry
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138, USA
ponbarry@eecs.harvard.edu

Stuart Shieber
School of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138, USA
shieber@seas.harvard.edu

Abstract

We present an experiment aimed at understanding how to optimally use acoustic and prosodic information to predict a speaker's level of certainty. With a corpus of utterances where we can isolate a single word or phrase that is responsible for the speaker's level of certainty we use different sets of sub-utterance prosodic features to train models for predicting an utterance's perceived level of certainty. Our results suggest that using prosodic features of the word or phrase responsible for the level of certainty and of its surrounding context improves the prediction accuracy without increasing the total number of features when compared to using only features taken from the utterance as a whole.

1 Introduction

Prosody is a fundamental part of human-to-human spoken communication; it can affect the syntactic and semantic interpretation of an utterance (Hirschberg, 2003) and it can be used by speakers to convey their emotional state. In recent years, researchers have found prosodic features to be useful in automatically detecting emotions such as annoyance and frustration (Ang et al., 2002) and in distinguishing positive from negative emotional states (Lee and Narayanan, 2005).

In this paper, we address the problem of predicting the perceived level of certainty of a spoken utterance. Specifically, we have a corpus of utterances where it is possible to isolate a single word or phrase responsible for the speaker's level of certainty. With this corpus we investigate whether using prosodic features of the word or phrase causing uncertainty and of its surrounding context improves the prediction accuracy when compared to using features taken only from the utterance as a whole.

This work goes beyond existing research by looking at the predictive power of prosodic features extracted from salient sub-utterance segments. Previous work on uncertainty has examined the predictive power of utterance- and intonational phrase-level prosodic features (Liscombe et al., 2005) as well as the relative strengths of correlations between level of certainty and sub-utterance prosodic features (Pon-Barry, 2008). Our results suggest that we can do a better job at predicting an utterance's perceived level of certainty by using prosodic features extracted from the whole utterance plus ones extracted from salient pieces of the utterance, without increasing the total number of features, than by using only features from the whole utterance.

This work is relevant to spoken language applications in which the system knows specific words or phrases that are likely to cause uncertainty. For example, this would occur in a tutorial dialogue system when the speaker answers a direct question (Pon-Barry et al., 2006; Forbes-Riley et al., 2008), or in language (foreign or ESL) learning systems and literacy systems (Alwan et al., 2007) when new vocabulary is being introduced.

2 Previous Work

Researchers have examined certainty in spoken language using data from tutorial dialogue systems (Liscombe et al., 2005) and data from an uncertainty corpus (Pon-Barry, 2008).

Liscombe et al. (2005) trained a decision tree

Proceedings of NAACL HLT 2009: Short Papers, pages 105–108,
Boulder, Colorado, June 2009.

classifier on utterance-level and intonational phrase-level prosodic features to distinguish between certain, uncertain, and neutral utterances. They achieved 76% accuracy, compared to a 66% accuracy baseline (choosing the most common class).

We have collected a corpus of utterances spoken under varying levels of certainty (Pon-Barry, 2008). The utterances were elicited by giving adult native English speakers a written sentence containing one or more gaps, then displaying multiple options for filling in the gaps and telling the speakers to read the sentence aloud with the gaps filled in according to domain-specific criteria. We elicited utterances in two domains: (1) using public transportation in Boston, and (2) choosing vocabulary words to complete a sentence. An example is shown below.

Q: How can I get from Harvard to the Silver Line?
A: Take the red line to _____
  a. South Station
  b. Downtown Crossing

The term 'context' refers to the fixed part of the response (*"Take the red line to _____",* in this example) and the term 'target word' refers to the word or phrase chosen to fill in the gap.

The corpus contains 600 utterances from 20 speakers. Each utterance was annotated for level of certainty, on a 5-point scale, by five human judges who listened to the utterances out of context. The average inter-annotator agreement (Kappa) was 0.45. We refer to the average of the five ratings as the 'perceived level of certainty' (the quantity we attempt to predict in this paper).

We computed correlations between perceived level of certainty and prosodic features extracted from the whole utterance, the context, and the target word. Pauses preceding the target word were considered part of the target word; all segmentation was done manually. Because the speakers had unlimited time to read over the context before seeing the target words, the target word is considered to be the *source* of the speaker's confidence or uncertainty; it corresponds to the decision that the speaker had to make. Our correlation results suggest that while some prosodic cues to level of certainty were strongest in the whole utterance, others were strongest in the context or the target word. In this paper, we extend this past work by testing the prediction accuracy of models trained on different subsets of these prosodic features.

3 Prediction Experiments

In our experiments we used 480 of the 600 utterances in the corpus, those which contained exactly one gap. (Some had two or three gaps.) We extracted the following 20 prosodic feature-types from each whole utterance, context, and target word (a total of 60 features) using WaveSurfer[1] and Praat[2].

Pitch: minf0, maxf0, meanf0, stdevf0, rangef0, relative position minf0, relative position maxf0, absolute slope (Hz), absolute slope (semitones)

Intensity: minRMS, maxRMS, meanRMS, stdevRMS, relative position minRMS, relative position maxRMS

Temporal: total silence, percent silence, total duration, speaking duration, speaking rate

These features are comparable to those used in Liscombe et al.'s (2005) prediction experiments. The pitch and intensity features were represented as z-scores normalized by speaker; the temporal features were not normalized.

Next, we created a 'combination' set of 20 features based on our correlation results. Figure 1 illustrates how the combination set was created: for each prosodic feature-type (each row in the table) we chose either the whole utterance feature, the context feature, or the target word feature, whichever one had the strongest correlation with perceived level of certainty. The selected features (highlighted in Figure 1) are listed below.

Whole Utterance: total silence, total duration, speaking duration, relative position maxf0, relative position maxRMS, absolute slope (Hz), absolute slope (semitones)

Context: minf0, maxf0, meanf0, stdevf0, rangef0, minRMS, maxRMS, meanRMS, relative position minRMS

Target Word: percent silence, speaking rate, relative position minf0, stdevRMS

[1] http://www.speech.kth.se/wavesurfer/
[2] http://www.fon.hum.uva.nl/praat/

Feature-type	Whole Utterance	Context	Target Word
min f0	0.107	**0.119**	0.041
max f0	−0.073	**−0.153**	−0.045
mean f0	0.033	**0.070**	−0.004
stdev f0	−0.035	**−0.047**	−0.043
range f0	−0.128	**−0.211**	−0.075
rel. position min f0	0.042	0.022	**0.046**
rel. position max f0	**0.015**	0.008	0.001
abs. slope f0 (Hz)	**0.275**	0.180	0.191
abs. slope f0 (Semi)	**0.160**	0.147	0.002
min RMS	0.101	**0.172**	0.027
max RMS	−0.091	**−0.110**	−0.034
mean RMS	−0.012	**0.039**	−0.031
stdev RMS	−0.002	−0.003	**−0.019**
rel. position min RMS	0.101	**0.172**	0.027
rel. position max RMS	**−0.039**	−0.028	−0.007
total silence	**−0.643**	−0.507	−0.495
percent silence	−0.455	−0.225	**−0.532**
total duration	**−0.592**	−0.502	−0.590
speaking duration	**−0.430**	−0.390	−0.386
speaking rate	0.090	0.014	**0.136**

Figure 1: *The* Combination *feature set (highlighted in table) was produced by selecting either the whole utterance feature, the context feature, or the target word feature for each prosodic feature-type, whichever one was most strongly correlated with perceived level of certainty.*

To compare the prediction accuracies of different subsets of features, we fit five linear regression models to the feature sets. The five subsets are: (A) whole utterance features only, (B) target word features only, (C) context features only, (D) all features, and (E) the combination feature set. We divided the data into 20 folds (one fold per speaker) and performed a 20-fold cross-validation for each set of features. Each experiment fits a model using data from 19 speakers and tests on the remaining speaker. Thus, when we test our models, we are testing the ability to classify utterances of an unseen speaker.

Table 1 shows the accuracies of the models trained on the five subsets of features. The numbers reported are averages of the 20 cross-validation accuracies. We report results for two cases: 5 prediction classes and 3 prediction classes. We first computed the prediction accuracy over five classes (the regression output was rounded to the nearest integer). Next, in order to compare our results to those of Liscombe et al. (2005), we recoded the 5-class results into 3-class results, following Pon-Barry (2008), in the way that maximized inter-annotator agreement. The naive baseline numbers are the accuracies that would be achieved by always choosing the most common class.

4 Discussion

Assuming that the target word is responsible for the speaker's level of certainty, it is not surprising that the target word feature set (B) yields higher accuracies than the context feature set (C). It is also not surprising that the set of all features (D) yields higher accuracies than sets (A), (B), and (C).

The key comparison to notice is that the combination feature set (E), with only 20 features, yields higher average accuracies than the utterance feature set (A): a difference of 6.42% for 5 classes and 5.83% for 3 classes. This suggests that using a combination of features from the context and target word in addition to features from the whole utterance leads to better prediction of the perceived level of certainty than using features from only the whole utterance.

One might argue that these differences are just due to noise. To address this issue, we compared the prediction accuracies of sets (A) and (E) per fold. This is illustrated in Figure 2. Each fold in our cross-validation corresponds to a different speaker, so the folds are *not* identically distributed and we do not expect each fold to yield the same prediction accuracy. That means that we should compare predictions of the two feature sets within folds rather than between folds. Figure 2 shows the correlations between the predicted and perceived levels of certainty for the models trained on sets (A) and (E). The combination set (E) predictions were more strongly correlated than whole utterance set (A) predictions in 16 out of 20 folds. This result supports our claim that using a combination of features from the context and target word in addition to features from the whole utterance leads to better prediction of level of certainty.

Our best prediction accuracy for the 3 class case, 74.79%, was slightly lower than the accuracy reported by Liscombe et al. (2005), 76.42%. However, our difference from the naive baseline was 18.54% where Liscombe et al.'s was 10.42%. Liscombe et al. randomly divided their data into training and test sets, so it is unclear whether they tested on seen or unseen speakers. Further, they ran one experiment rather than a cross-validation, so their reported accuracy may not be indicative of the entire data set.

We also trained support vector models on these subsets of features. The main result was the same:

Table 1: *Average prediction accuracies for the linear regression models trained on five subsets of prosodic features. The models trained on the* Combination *feature set and the* All *feature set perform better than the other three models in both the 3- and 5-class settings.*

Feature Set	Num Features	Accuracy (5 classes)	Accuracy (3 classes)
Naive Baseline	N/A	31.46%	56.25%
(A) Utterance	20	39.00%	68.96%
(B) Target Word	20	43.13%	68.96%
(C) Context	20	37.71%	67.50%
(D) All	60	48.54%	74.58%
(E) Combination	20	45.42%	74.79%

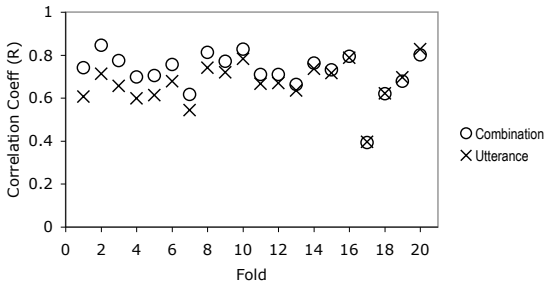


Figure 2: *Correlations with perceived level of certainty per fold for the* Combination (O) *and the* Utterance (X) *feature set predictions, sorted by the size of the difference. In 16 of the 20 experiments, the correlation coefficients for the* Combination *feature set are greater than those of the* Utterance *feature set.*

the set of all features (D) and the combination set (E) had better prediction accuracies than the utterance feature set (A). In addition, the combination set (E) had the best prediction accuracies (of all models) in both the 3- and 5-class settings. The raw accuracies were approximately 5% lower than those of the linear regression models.

5 Conclusion and Future Work

The results of our experiments suggest a better predictive model of level of certainty for systems where words or phrases likely to cause uncertainty are known ahead of time. Without increasing the total number of features, combining select prosodic features from the target word, the surrounding context and the whole utterance leads to better prediction of level of certainty than using features from the whole utterance only. In the near future, we plan to experiment with prediction models of the speaker's self-reported level of certainty.

Acknowledgments

This work was supported by a National Defense Science and Engineering Graduate Fellowship.

References

Abeer Alwan, Yijian Bai, Matthew Black, et al. 2007. A system for technology based assessment of language and literacy in young children: the role of multiple information sources. *Proc. of IEEE International Workshop on Multimedia Signal Processing*, pp. 26–30, Chania, Greece.

Jeremy Ang, Rajdip Dhillon, Ashley Krupski, et al. 2002. Prosody-based automatic detection of annoyance and frustration in human-computer dialog. *Proc. of ICSLP 2002*, pp. 2037–2040, Denver, CO.

Kate Forbes-Riley, Diane Litman, and Mihai Rotaru. 2008. Responding to student uncertainty during computer tutoring: a preliminary evaluation. *Proc. of the 9th International Conference on Intelligent Tutoring Systems*, Montreal, Canada.

Julia Hirschberg. 2003. Intonation and pragmatics. In L. Horn and G. Ward (ed.), *Handbook of Pragmatics*, Blackwell.

Chul Min Lee and Shrikanth Narayanan. 2005. Towards detecting emotions in spoken dialogs. *IEEE Transactions on Speech and Audio Processing*, 13(2):293–303.

Jackson Liscombe, Julia Hirschberg, and Jennifer Venditti. 2005. Detecting certainness in spoken tutorial dialogues. *Proceedings of Eurospeech 2005*, Lisbon, Portugal.

Heather Pon-Barry, Karl Schultz, Elizabeth Bratt, Brady Clark, and Stanley Peters. 2006. Responding to student uncertainty in spoken tutorial dialogue systems. *International Journal of Artificial Intelligence in Education* 16:171-194.

Heather Pon-Barry. 2008. Prosodic manifestations of confidence and uncertainty in spoken language. *Proc. of Interspeech 2008*, pp. 74–77, Brisbane, Australia.

Using Integer Linear Programming for Detecting Speech Disfluencies

Kallirroi Georgila
Institute for Creative Technologies, University of Southern California
13274 Fiji Way, Marina del Rey, CA 90292, USA
kgeorgila@ict.usc.edu

Abstract

We present a novel two-stage technique for detecting speech disfluencies based on Integer Linear Programming (ILP). In the first stage we use state-of-the-art models for speech disfluency detection, in particular, hidden-event language models, maximum entropy models and conditional random fields. During testing each model proposes possible disfluency labels which are then assessed in the presence of local and global constraints using ILP. Our experimental results show that by using ILP we can improve the performance of our models with negligible cost in processing time. The less training data is available the larger the improvement due to ILP.

1 Introduction

Speech disfluencies (also known as speech repairs) occur frequently in spontaneous speech and can pose difficulties to natural language processing (NLP) since most NLP tools (e.g. parsers, part-of-speech taggers, information extraction modules) are traditionally trained on written language. Speech disfluencies can be divided into three intervals, the *reparandum*, the *editing term* and the *correction* (Heeman and Allen, 1999; Liu et al., 2006).

```
(it was) * (you know) it was set
```

In the above example, "it was" is the reparandum, "you know" is the editing term and the remaining sentence is the correction. The asterisk marks the interruption point at which the speaker halts the original utterance in order to start the repair. The editing term is optional and consists of one or more filled pauses (e.g. uh, uh-huh) or discourse markers (e.g. you know, so). Some researchers include editing terms in the definition of disfluencies. Here we focus only on detecting repetitions (the speaker repeats some part of the utterance), revisions (the speaker modifies the original utterance) or restarts (the speaker abandons an utterance and starts over). We also deal with complex disfluencies, i.e. a series of disfluencies in succession ("I think I think uh I believe that...").

In previous work many different approaches to detecting speech disfluencies have been proposed. Different types of features have been used, e.g. lexical features only, acoustic and prosodic features only or a combination of both (Liu et al., 2006). Furthermore, a number of studies have been conducted on human transcriptions while other efforts have focused on detecting disfluencies from the speech recognition output.

In this paper we propose a novel framework for speech disfluency detection based on Integer Linear Programming (ILP). With Linear Programming (LP) problems the goal is to optimize a linear objective function subject to linear equality and linear inequality constraints. When some or all the variables of the objective function and the constraints are non-negative integers, LP becomes ILP. ILP has recently attracted much attention in NLP. It has been applied to several problems including sentence compression (Clarke and Lapata, 2008) and relation extraction (Roth and Yih, 2004). Some of these methods (e.g. (Roth and Yih, 2004)) follow the two-stage approach of first hypothesizing a list of possible answers using a classifier and then selecting the best answer by applying ILP. We have adopted this two-stage approach and applied it to speech disfluency detection.

In the first stage we use state-of-the-art tech-

Proceedings of NAACL HLT 2009: Short Papers, pages 109–112,
Boulder, Colorado, June 2009.

niques for speech disfluency detection, in particular, Hidden-Event Language Models (HELMs) (Stolcke and Shriberg, 1996), Maximum Entropy (ME) models (Ratnaparkhi, 1998) and Conditional Random Fields (CRFs) (Lafferty et al., 2001). Nevertheless, any other classification method could be used instead. During testing each classifier proposes possible labels which are then assessed in the presence of local and global constraints using ILP. ILP makes the final decision taking into account both the constraints and the output of the classifier.

In the following we use the Switchboard corpus and only lexical features for training our 3 classifiers. Then we apply ILP to the output of each classifier. Our goal is not to investigate the best set of features or achieve the best possible results. In that case we could also use prosodic features as they have been shown to improve performance. Our target is to show that by using ILP we can improve with negligible cost in processing time the performance of state-of-the-art techniques, especially when not much training data is available.

The novelty of our work lies in the two following areas: First, we propose a novel approach for detecting disfluencies with improvements over state-of-the-art models (HELMs, ME models and CRFs) that use similar lexical features. Although the two-stage approach is not unique, as discussed above, the formulation of the ILP objective function and constraints for disfluency detection is entirely novel. Second, we compare our models using the tasks of both detecting the interruption point and finding the beginning of the reparandum. In previous work (Liu et al., 2006) Hidden Markov Models (combination of decision trees and HELMs) and ME models were trained to detect the interruption points and then heuristic rules were applied to find the correct onset of the reparandum in contrast to CRFs that were trained to detect both points at the same time.

The structure of the paper is as follows: In section 2 we describe our data set. In section 3 we describe our approach in detail. Then in section 4 we present our experiments and provide results. Finally in section 5 we present our conclusion and propose future work.

2 Data Set

We use Switchboard (LDC catalog LDC99T42), which is traditionally used for speech disfluency experiments. We transformed the Switchboard annotations into the following format:

```
I BE was IE one IP I was right
```

BE (beginning of edit) is the point where the reparandum starts and IP is the interruption point (the point before the repair starts). In the above example the beginning of the reparandum is the first occurrence of "I", the interruption point appears after "one" and every word between BE and IP is tagged as IE (inside edit). Sometimes BE and IP occur at the same point, e.g. "I BE-IP I think".

The number of occurrences of BE and IP in our training set are 34387 and 39031 respectively, in our development set 3146 and 3499, and in our test set 6394 and 7413.

3 Methodology

In the first stage we train our classifier. Any classifier can be used as long as it provides more than one possible answer (i.e. tag) for each word in the utterance. Valid tags are BE, BE-IP, IP, IE or O. The O tag indicates that the word is outside the disfluent part of the utterance. ILP will be applied to the output of the classifier during testing.

Let N be the number of words of each utterance and i the location of the word in the utterance (i=1,...,N). Also, let $C_{BE}(i)$ be a binary variable (1 or 0) for the BE tag. Its value will be determined by ILP. If it is 1 then the word will be tagged as BE. In the same way, we use $C_{BE-IP}(i)$, $C_{IP}(i)$, $C_{IE}(i)$, $C_O(i)$ for tags BE-IP, IP, IE and O respectively. Let $P_{BE}(i)$ be the probability given by the classifier that the word is tagged as BE. In the same way, let $P_{BE-IP}(i)$, $P_{IP}(i)$, $P_{IE}(i)$, $P_O(i)$ be the probabilities for tags BE-IP, IP, IE and O respectively. Given the above definitions, the ILP problem formulation can be as follows:

$$max[\sum_{i=1}^{N}[P_{BE}(i)C_{BE}(i) + P_{BE-IP}(i)C_{BE-IP}(i) + P_{IP}(i)C_{IP}(i) + P_{IE}(i)C_{IE}(i) + P_O(i)C_O(i)]] \quad (1)$$

subject to:

$$C_{BE}(i) + C_{BE-IP}(i) + C_{IP}(i) + C_{IE}(i) + C_O(i) = 1 \qquad \forall i \in (1, ..., N) \quad (2)$$

$$C_{BE}(1) + C_{BE-IP}(1) + C_O(1) = 1 \quad (3)$$

$$C_{BE-IP}(N) + C_{IP}(N) + C_O(N) = 1 \quad (4)$$

$$C_{BE}(i) - C_{BE-IP}(i-1) - C_{IP}(i-1) - C_O(i-1) \leq 0 \qquad \forall i \in (2, ..., N) \quad (5)$$

$$1 - C_{BE}(i) - C_{BE}(i-1) \geq 0 \quad \forall i \in (2, ..., N) \quad (6)$$

Equation 1 is the linear objective function that we want to maximize, i.e. the overall probability of the utterance. Equation 2 says that each word can have one tag only. Equation 3 denotes that the first word is either BE, BE-IP or O. Equation 4 says that the last word is either BE-IP, IP or O. For example the last word cannot be BE because then we would expect to see an IP. Equation 5 defines the transitions that are allowed between tags as described in Table 1 (first row). Equation 5 says that if we have a word tagged as BE it means that the previous word was tagged as BE-IP or IP or O. It could not have been tagged as IE because IE must be followed by an IP before a new disfluency starts. Also, it could not have been BE because then we would expect to see an IP. From Table 1 we can easily define 4 more equations for the rest of the tags. Finally, equation 6 denotes that we cannot transition from BE to BE (we need an IP in between).

We also formulate some additional rules that describe common disfluency patterns. First, let us have an example of a long-context rule. If we have the sequence of words "he was the one um you know she was the one", we expect this to be tagged as "he BE was IE the IE one IP um O you O know O she O was O the O one O", if we do not take into account the context in which this pattern occurs. We incorporate this rule into our ILP problem formulation as follows: Let (w_1,...,w_N) be a sequence of N words where both w_2 and w_{N-7} are personal pronouns, the word sequence w_3,w_4,w_5 is the same as the sequence w_{N-6},w_{N-5},w_{N-4} and all the words in between (w_6,...,w_{N-8}) are filled pauses or discourse markers. Then the probabilities given by the classifier are modified as follows: $P_{BE}(2)$=$P_{BE}(2)$+$th1$, $P_{IE}(3)$=$P_{IE}(3)$+$th2$, $P_{IE}(4)$=$P_{IE}(4)$+$th3$ and $P_{IP}(5)$=$P_{IP}(5)$+$th4$, where $th1$, $th2$, $th3$ and $th4$ are empirically set thresholds (between 0.5 and 1, using the development set of the corpus).

Now, here is an example of a short-context rule. If we have the same word appear 3 times in a row ("do do do") we expect this to be tagged as "do BE-IP do IP do O". To incorporate this rule into our ILP problem formulation we can modify the probabilities given by the classifier accordingly.

In total we have used 7 rules that deal with short-context and 5 rules that deal with long-context dependencies. From now on we will refer to the model that uses all rules (general ILP formulation and all pattern-based rules) as ILP and to the model that uses only the general ILP constraints and the short-context pattern-based rules as ILP-. In all rules, we can skip editing terms (see example above).

From Tag	To Tag
BE-IP or IP or O	BE
BE-IP or IP or O	BE-IP
BE or BE-IP or IP or IE	IP
BE or BE-IP or IP or IE	IE
BE-IP or IP or O	O

Table 1: Possible transitions between tags.

4 Experiments

For HELMs we use the SRI Statistical Language Modeling Toolkit. Each utterance is a sequence of word and Part-of-Speech (POS) pairs fed into the toolkit: `i/prp BE was/vbd IE one/cd IP i/prp was/vbd right/jj`. We report results with 4-grams. For ME we use the OpenNLP MaxEnt toolkit and for CRFs the toolkit CRF++ (both available from `sourceforge`). We experimented with different sets of features and we achieved the best results with the following setup (i is the location of the word or POS in the sentence): Our word features are $\langle w_i \rangle$, $\langle w_{i+1} \rangle$, $\langle w_{i-1}, w_i \rangle$, $\langle w_i, w_{i+1} \rangle$, $\langle w_{i-2}, w_{i-1}, w_i \rangle$, $\langle w_i, w_{i+1}, w_{i+2} \rangle$. Our POS features have the same structure as the word features. For ILP we use the `lp_solve` software also available from `sourceforge`.

For evaluating the performance of our models we use standard metrics proposed in the literature, i.e. F-score and NIST Error Rate. We report results for BE and IP. F-score is the harmonic mean of precision and recall (we equally weight precision and recall). Precision is computed as the ratio of the correctly identified tags X to all the tags X detected by the model (where X is BE or IP). Recall is the ratio of the correctly identified tags X to all the tags X that appear in the reference utterance. The NIST Error Rate measures the average number of incorrectly identified tags per reference tag, i.e. the sum of insertions, deletions and substitutions divided by the total number of reference tags (Liu et al., 2006). To calculate the level of statistical significance we always use the Wilcoxon signed-rank test.

Table 2 presents comparative results between our models. The ILP and ILP- models lead to significant improvements compared to the plain models for HELMs and ME ($p<10^{-8}$, plain models vs. ILP and ILP-). With CRFs the improvement is smaller,

	BE		IP	
	F-score	**Error**	**F-score**	**Error**
4gram	60.3	54.8	67.0	50.7
4gram ILP	**76.0**	**38.1**	**79.0**	**38.0**
4gram ILP-	73.9	39.5	77.9	38.3
ME	63.8	52.6	72.8	44.3
ME ILP	**77.9**	**36.3**	80.8	35.4
ME ILP-	75.6	37.2	**81.0**	**33.7**
CRF	78.6	34.3	82.0	31.7
CRF ILP	**80.1**	34.5	82.5	33.3
CRF ILP-	79.8	**33.5**	**83.4**	**30.5**

Table 2: Comparative results between our models.

	25%	**50%**	**75%**	**100%**
4gram	59.8	56.6	56.2	54.8
4gram ILP	40.2	38.9	38.2	38.0
4gram ILP-	42.1	40.7	39.8	39.5
ME	61.6	56.9	54.7	52.6
ME ILP	38.5	37.7	36.5	36.3
ME ILP-	39.7	38.7	37.6	37.2
CRF	40.3	37.1	35.5	34.3
CRF ILP	37.1	36.2	35.2	34.5
CRF ILP-	36.6	35.5	34.4	33.5

Table 3: Error rate variation for BE depending on the training set size.

$p<0.03$ (CRF vs. CRF with ILP), not significant (CRF vs. CRF with ILP-), $p<0.0008$ (CRF with ILP vs. CRF with ILP-). HELMs and ME models benefit more from the ILP model than the ILP- model (ME only for the BE tag) whereas ILP- appears to perform better than ILP for CRFs.

Table 3 shows the effect of the training set size on the error rates only for BE due to space restrictions. The trend is similar for IP. The test set is always the same. Both ILP and ILP- perform better than the plain models. This is true even when the ILP and ILP- models are trained with less data (HELMs and ME models only). Note that HELM (or ME) with ILP or ILP- trained on 25% of the data performs better than plain HELM (or ME) trained on 100% of the data ($p<10^{-8}$). This is very important because collecting and annotating data is expensive and time-consuming. Furthermore, for CRFs in particular the training process takes long especially for large data sets. In our experiments CRFs took about 400 iterations to converge (approx. 136 min for the whole training set) whereas ME models took approx. 48 min for the same number of iterations and training set size. Also, ME models trained with 100 iterations (approx. 11 min) performed better than ME models trained with 400 iterations. The cost of applying ILP is negligible since the process is fast and applied during testing.

5 Conclusion

We presented a novel two-stage technique for detecting speech disfluencies based on ILP. In the first stage we trained HELMs, ME models and CRFs. During testing each classifier proposed possible labels which were then assessed in the presence of local and global constraints using ILP. We showed that ILP can improve the performance of state-of-the-art classifiers with negligible cost in processing time, especially when not much training data is available. The improvement is significant for HELMs and ME models. In future work we will experiment with acoustic and prosodic features and detect disfluencies from the speech recognition output.

Acknowledgments

This work was sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). The content does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred.

References

J. Clarke and M. Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 31:399–429.

P. Heeman and J. Allen. 1999. Speech repairs, intonational phrases and discourse markers: Modeling speakers' utterances in spoken dialogue. *Computational Linguistics*, 25:527–571.

J. Lafferty, A. McCallum, and F. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*.

Y. Liu, E. Shriberg, A. Stolcke, D. Hillard, M. Ostendorf, and M. Harper. 2006. Enriching speech recognition with automatic detection of sentence boundaries and disfluencies. *IEEE Trans. Audio, Speech and Language Processing*, 14(5):1526–1540.

A. Ratnaparkhi. 1998. *Maximum Entropy Models for natural language ambiguity resolution*. Ph.D. thesis, University of Pennsylvania.

D. Roth and W. Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proc. of CoNNL*.

A. Stolcke and E. Shriberg. 1996. Statistical language modeling for speech disfluencies. In *Proc. of ICASSP*.

Contrastive Summarization: An Experiment with Consumer Reviews

Kevin Lerman
Columbia University
New York, NY
klerman@cs.columbia.edu

Ryan McDonald
Google Inc.
New York, NY
ryanmcd@google.com

Abstract

Contrastive summarization is the problem of jointly generating summaries for two entities in order to highlight their differences. In this paper we present an investigation into contrastive summarization through an implementation and evaluation of a contrastive opinion summarizer in the consumer reviews domain.

1 Introduction

Automatic summarization has historically focused on summarizing events, a task embodied in the series of Document Understanding Conferences[1]. However, there has also been work on *entity-centric summarization*, which aims to produce summaries from text collections that are relevant to a particular entity of interest, e.g., product, person, company, etc. A well-known example of this is from the opinion mining community where there has been a number of studies on summarizing the expressed sentiment towards entities (cf. Hu and Liu (2006)). Another recent example of entity-centric summarization is the work of Filippova et al. (2009) to produce company-specific financial report summaries.

In this study we investigate a variation of entity-centric summarization where the goal is not to summarize information about a single entity, but pairs of entities. Specifically, our aim is to jointly generate two summaries that highlight differences between the entities – a task we call *contrastive summarization*. An obvious application comes from the consumer reviews domain, where a person considering a purchase wishes to see the differences in opinion about the top candidates without reading all the reviews for each product. Other applications include contrasting financial news about related companies or comparing platforms of political candidates.

Contrastive summarization has many points of comparison in the NLP, IR and Data-Mining literature. Jindal and Liu (2006) introduce techniques to find and analyze explicit comparison sentences, but this assumes that such sentences exist. In contrastive summarization, there is no assumption that two entities have been explicitly compared. The goal is to automatically generate the comparisons based on the data. In the IR community, Sun et al. (2006) explores retrieval systems that align query results to highlight points of commonality and difference. In contrast, we attempt to identify contrasts from the data, and then generate summaries that highlight them. The *novelty detection task* of determining whether a new text in a collection contains information distinct from that already gathered is also related (Soboroff and Harman, 2005). The primary difference here is that contrastive summarization aims to extract information from one collection not present in the other in addition to information present in both collections that highlights a difference between the entities.

This paper describes a contrastive summarization experiment where the goal is to generate contrasting opinion summaries of two products based on consumer reviews of each. We look at model design choices, describe an implementation of a contrastive summarizer, and provide an evaluation demonstrating a significant improvement in the usefulness of contrastive summaries versus summaries generated by single-product opinion summarizers.

2 Single-Product Opinion Summarization

As input we assume a set of relevant text excerpts (typically sentences), $T = \{t_1, \ldots, t_m\}$, which con-

[1] http://duc.nist.gov/

Proceedings of NAACL HLT 2009: Short Papers, pages 113–116,
Boulder, Colorado, June 2009.

tain opinions about some product of interest. The goal of opinion summarization[2] is to select some number of text excerpts to form a summary S of the product so that S is representative of the average opinion and speaks to its important aspects (also proportional to opinion), which we can formalize as:

$$S = \arg\max_{S \subseteq T} \mathcal{L}(S) \quad \text{s.t. } \text{LENGTH}(S) \leq K$$

where $\mathcal{L}$ is some score over possible summaries that embodies what a user might desire in an opinion summary, $\text{LENGTH}(S)$ is the length of the summary and K is a pre-specified length constraint.

We assume the existence of standard sentiment analysis tools to provide the information used in the scoring function $\mathcal{L}$. First, we assume the tools can assign a sentiment score from -1 (negative) to 1 (positive) to an arbitrary span of text. Second, we assume that we can extract a set of aspects that the text is discussing (e.g, "The sound was crystal clear" is about the aspect *sound quality*). We refer the reader to abundance of literature on sentiment analysis for more details on how such tools can be constructed (cf. Pang and Lee (2008)). For this study, we use the tools described and evaluated in Lerman et al. (2009). We note however, that the subject of this discussion is not the tools themselves, but their use.

The single product opinion summarizer we consider is the Sentiment Aspect Match model (SAM) described and evaluated in (Lerman et al., 2009). Underlying SAM is the assumption that opinions can be described by a bag-of-aspects generative process where each aspect is generated independently and the sentiment associated with the aspect is generated conditioned on its identity,

$$p(t) = \prod_{a \in A_t} p(a) p(\text{SENT}(a_t)|a)$$

where A_t is a set of aspects that are mentioned in text excerpt t, $p(a)$ is the probability of seeing aspect a, and $\text{SENT}(a_t) \in [-1, 1]$ is the sentiment associated with aspect a in t. The SAM model sets $p(a)$ through the maximum likelihood estimates over T and assumes $p(\text{SENT}(a_t)|a)$ is normally distributed with a mean and variance also estimated from T. We denote $\text{SAM}(T)$ as the model learned using the entire set of candidate text excerpts T.

The SAM summarizer scores each potential summary, S, by learning another model $\text{SAM}(S)$ based on the text excerpts used to construct S. We can then measure the distance between a model learned over the full set T and a summary S by summing the KL-divergence between their learned probability distributions. In our case we have $1 + |A_T|$ distributions – $p(a)$, and $p(\cdot|a)$ for all $a \in A_T$. We then define $\mathcal{L}$:

$$\mathcal{L}(S) = -\text{KL}(\text{SAM}(T), \text{SAM}(S))$$

That is, the SAM summarizer prefers summaries whose induced model is close to the model induced for all the opinions about the product of interest. Thus, a good summary should (1) mention aspects in roughly the same proportion that they are mentioned in the full set of opinions *and* (2) mention aspects with sentiment also in proportion to what is observed in the full opinion set. A high scoring summary is found by initializing a summary with random sentences and hill-climbing by replacing sentences one at a time until convergence.

We chose to use the SAM model for our experiment for two reasons. First, Lerman et al. (2009) showed that among a set of different opinion summarizers, SAM was rated highest in a user study. Secondly, as we will show in the next section, the SAM summarization model can be naturally extended to produce contrastive summaries.

3 Constrastive Summarization

When jointly generating pairs of summaries, we attempt to highlight differences between two products. These differences can take multiple forms. Clearly, two products can have different prevailing sentiment scores with respect to an aspect (e.g. "Product X has great image quality" vs "Product Y's image quality is terrible"). Reviews of different products can also emphasize different aspects. Perhaps one product's screen is particularly good or bad, but another's is not particularly noteworthy – or perhaps the other product simply doesn't have a screen. Regardless of sentiment, reviews of the first product will emphasize the screen quality aspect more than those of the second, indicating that our summary should as well.

[2]We focus on text-only opinion summaries as opposed to those based on numeric ratings (Hu and Liu, 2006).

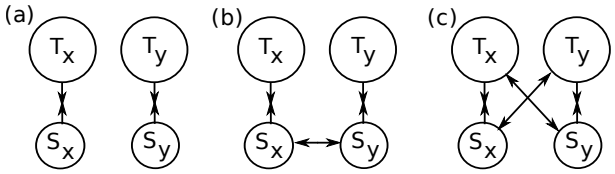


Figure 1: (a) Non-joint model: Generates summaries for two products independently. (b) Joint model: Summaries attempt to look like text they are drawn from, but contrast each-other. (c) Joint model: Like (b), except summaries contrast text that the other summary is drawn from.

As input to our contrastive summarizer we assume two products, call them x and y as well as two corresponding candidate sets of opinions, T_x and T_y, respectively. As output, a contrastive summarizer will produce two summaries – S_x for product x and S_y for product y – so that the summaries highlight the differences in opinion between the two products.

What might a contrastive summarizer look like on a high-level? Figure 1 presents some options. The first example (1a) shows a system where each summary is generated independently, i.e., running the SAM model on each product separately without regard to the other. This procedure may provide some useful contrastive information, but any such information will be present incidentally. To make the summaries specifically contrast each other, we can modify our system by explicitly modeling the fact that we want summaries S_x and S_y to contrast. In the SAM model this is trivial as we can simply add a term to the scoring function $\mathcal{L}$ that attempts to maximize the KL-divergence between the two summaries induced models $\text{SAM}(S_x)$ and $\text{SAM}(S_y)$.

This approach is graphically depicted in figure 1b, where the system attempts to produce summaries that are maximally similar to the opinion set they are drawn from and minimally similar from each other. However, some obvious degenerate solutions arise if we chose to model our system this way. Consider two products, x and y, for which all opinions discuss two aspects a and b with identical frequency and sentiment polarity. Furthermore, several opinions of x and y discuss an aspect c, but with opposite sentiment polarity. Suppose we have to build contrastive summaries and only have enough space to cover a single aspect. The highest scoring contrastive pair of summaries would consist of one for x that mentions a exclusively, and one for y that mentions b exclusively – these summaries each mention a promiment aspect of their product, and have no overlap with each other. However, they provide a false contrast because they each attempt to contrast the other *s*ummary, rather than the other *p*roduct. Better would be for both to cover aspect c.

To remedy this, we reward summaries that instead have a high KL-divergence with respect to the other product's *full* model $\text{SAM}(T)$ as depicted in Figure 1c. Under this setup, the degenerate solution described above is no longer appealing, as both summaries have the same KL-divergence with respect to the other product as they do to their own product. The fact that the summaries themselves are dissimilar is irrelevant. Comparing the summaries only to the products' full language models prevents us from rewarding summaries that convey a false contrast between the products under comparison. Specifically, we now optimize the following joint summary score:

$$\begin{aligned}\mathcal{L}(S_x, S_y) \quad = \quad & -\text{KL}(\text{SAM}(T_x), \text{SAM}(S_x)) \\ & -\text{KL}(\text{SAM}(T_y), \text{SAM}(S_y)) \\ & +\text{KL}(\text{SAM}(T_x), \text{SAM}(S_y)) \\ & +\text{KL}(\text{SAM}(T_y), \text{SAM}(S_x))\end{aligned}$$

Note that we could additionally model divergence between the two summaries (i.e., merging models in figures 1b and c), but such modeling is redundant. Furthermore, by not explicitly modeling divergence between the two summaries we simplify the search space as each summary can be constructed without knowledge of the content of the second summary.

4 The Experiment

Our experiments focused on consumer electronics. In this setting an entity to be summarized is one specific product and T is a set of segmented user reviews about that product. We gathered reviews for 56 electronics products from several sources such as CNet, Epinions, and PriceGrabber. The products covered 15 categories of electronics products, including MP3 players, digital cameras, laptops, GPS systems, and more. Each had at least four reviews, and the mean number of reviews per product was 70.

We manually grouped the products into categories (MP3 players, cameras, printers, GPS sys-

System	As Received	Consolidated
SAM	1.85 $\pm$ 0.05	1.82 $\pm$ 0.05
SAM + contrastive	**1.76** $\pm$ 0.05	**1.68** $\pm$ 0.05

Table 1: Mean rater scores for contrastive summaries by system. Scores range from 0-3 and lower is better.

System	2+ raters	All 3 raters
SAM	0.8	0.2
SAM + contrastive	**2.0**	**0.6**

Table 2: Average number of points of contrast per comparison observed by multiple raters, by system. Raters were asked to list up to 3. Higher is better.

tems, headphones, computers, and others), and generated contrastive summaries for each pair of products in the same category using 2 different algorithms: (1) The SAM algorithm for each product individually (figure 1a) and (2) The SAM algorithm with our adaptation for contrastive summarization (figure 1c). Summaries were generated using $K = 650$, which typically consisted of 4 text excerpts of roughly 160 characters. This allowed us to compare different summaries without worrying about the effects of summary length on the ratings. In all, we gathered 178 contrastive summaries (89 per system) to be evaluated by raters and each summary was evaluated by 3 random raters resulting in 534 ratings. The raters were 55 everyday internet users that signed-up for the experiment and were assigned roughly 10 random ratings each. Raters were shown two products and their contrastive summaries, and were asked to list 1-3 differences between the products as seen in the two summaries. They were also asked to read the products' reviews to help ensure that the differences observed were not simply artifacts of the summarizer but in fact are reflected in actual opinions. Finally, raters were asked to rate the helpfulness of the summaries in identifying these distinctions, rating each with an integer score from 0 ("extremely useful") to 3 ("not useful").

Upon examining the results, we found that raters had a hard time finding a meaningful distinction between the two middle ratings of 1 and 2 ("useful" and "somewhat useful"). We therefore present two sets of results: one with the scores as received from raters, and another with all 1 and 2 votes consolidated into a single class of votes with numerical score 1.5. Table 1 gives the average scores per system, lower scores indicating superior performance.

5 Analysis and Conclusions

The scores indicate that the addition of the contrastive term to the SAM model improves helpfulness, however both models roughly have average scores in the somewhat-useful to useful range. The difference becomes more pronounced when looking at the consolidated scores. The natural question arises: does the relatively small increase in helpfulness reflect that the contrastive summarizer is doing a poor job? Or does it indicate that users only find slightly more utility in contrastive information in this domain? We inspected comments left by raters in an attempt to answer this. Roughly 80% of raters were able to find at least two points of contrast in summaries generated by the SAM+contrastive versus 40% for summaries generated by the simple SAM model. We then examined the consistency of rater comments, i.e., to what degree did different raters identify the same points of contrast from a specific comparison? We report the results in table 2. Note that by this metric in particular, the contrastive summarizer outperforms its the single-product summarizer by significant margins and provides a strong argument that the contrastive model is doing its job.

Acknowledgements: The Google sentiment analysis team for insightful discussions and suggestions.

References

K. Filippova, M. Surdeanu, M. Ciaramita, and H. Zaragoza. 2009. Company-oriented extractive summarization of financial news. In *Proc. EACL*.

M. Hu and B. Liu. 2006. Opinion extraction and summarization on the web. In *Proc. AAAI*.

N. Jindal and B. Liu. 2006. Mining comparative sentences and relations. In *Proc. AAAI*.

Kevin Lerman, Sasha Blair-Goldensohn, and Ryan McDonald. 2009. Sentiment summarization: Evaluating and learning user preferences. In *Proc. EACL*.

B. Pang and L. Lee. 2008. *Opinion mining and sentiment analysis*. Now Publishers.

I. Soboroff and D. Harman. 2005. Novelty detection: The TREC experience. In *Proc. HLT/EMNLP*.

Sun, Wang, Shen, Zeng, and Chen. 2006. CWS: A Comparative Web search System. In *Proc. WWW*.

Topic Identification Using Wikipedia Graph Centrality

Kino Coursey
University of North Texas and Daxtron Laboratories, Inc.
kino@daxtron.com

Rada Mihalcea
University of North Texas
rada@cs.unt.edu

Abstract

This paper presents a method for automatic topic identification using a graph-centrality algorithm applied to an encyclopedic graph derived from Wikipedia. When tested on a data set with manually assigned topics, the system is found to significantly improve over a simpler baseline that does not make use of the external encyclopedic knowledge.

1 Introduction

Document topics have been used for a long time by librarians to improve the retrieval of a document, and to provide background or associated information for browsing by users. They can also assist search, background information gathering and contextualization tasks, and enhanced relevancy measures.

The goal of the work described in this paper is to automatically find topics that are relevant to an input document. We refer to this task as "topic identification" (Medelyan and Witten, 2008). For instance, starting with a document on "United States in the Cold War," we want to identify relevant topics, such as "history," "Global Conflicts," "Soviet Union," and so forth. We propose an unsupervised method for topic identification, based on a biased graph centrality algorithm applied to a large knowledge graph built from Wikipedia.

The task of topic identification goes beyond keyword extraction, since relevant topics may not be necessarily mentioned in the document, and instead have to be obtained from some repositories of external knowledge. The task is also different from text classification, since the topics are either not known in advance or are provided in the form of a controlled vocabulary with thousands of entries, and thus no classification can be performed. Instead, with topic identification, we aim to find topics (or categories[1]) that are relevant to the document at hand, which can be used to enrich the content of the document with relevant external knowledge.

2 Dynamic Ranking of Topic Relevance

Our method is based on the premise that external encyclopedic knowledge can be used to identify relevant topics for a given document.

The method consists of two main steps. In the first step, we build a knowledge graph of encyclopedic concepts based on Wikipedia, where the nodes in the graph are represented by the entities and categories that are defined in this encyclopedia. The edges between the nodes are represented by their relation of proximity inside the Wikipedia articles. The graph is built once and then it is stored offline, so that it can be efficiently use for the identification of topics in new documents.

In the second step, for each input document, we first identify the important encyclopedic concepts in the text, and thus create links between the content of the document and the external encyclopedic graph. Next, we run a biased graph centrality algorithm on the entire graph, so that all the nodes in the external knowledge repository are ranked based on their relevance to the input document.

2.1 Wikipedia

Wikipedia (http://en.wikipedia.org) is a free online encyclopedia, representing the outcome of a continuous collaborative effort of a large number of volunteer contributors. The basic entry is an *article*, which defines an entity or an event, and consists of a hypertext document with hyperlinks to other pages within or outside Wikipedia. In addition to arti-

[1]Throughout the paper, we use the terms "topic" and "category" interchangeably.

Proceedings of NAACL HLT 2009: Short Papers, pages 117–120,
Boulder, Colorado, June 2009.

cles, Wikipedia also includes a large number of categories, which represent topics that are relevant to a given article (the July 2008 version of Wikipedia includes more than 350,000 such categories).

We use the entire English Wikipedia to build an encyclopedic graph for use in the topic identification process. The nodes in the graph are represented by all the article and category pages in Wikipedia, and the edges between the nodes are represented by their relation of proximity inside the articles. The graph contains 5.8 million nodes, and 65.5 million edges.

2.2 Wikify!

In order to automatically identify the important encyclopedic concepts in an input text, we use the unsupervised system Wikify! (Mihalcea and Csomai, 2007), which identifies the concepts in the text that are likely to be highly relevant for the input document, and links them to Wikipedia concepts.

Wikify! works in three steps, namely: (1) candidate extraction, (2) keyword ranking, and (3) word sense disambiguation. The candidate extraction step parses the input document and extracts all the possible n-grams that are also present in the vocabulary used in the encyclopedic graph (i.e., anchor texts for links inside Wikipedia or article or category titles).

Next, the ranking step assigns a numeric value to each candidate, reflecting the likelihood that a given candidate is a valuable keyword. Wikify! uses a "keyphraseness" measure to estimate the probability of a term W to be selected as a keyword in a document by counting the number of documents where the term was already selected as a keyword $count(D_{key})$ divided by the total number of documents where the term appeared $count(D_W)$. These counts are collected from all the Wikipedia articles.

$$P(keyword|W) \approx \frac{count(D_{key})}{count(D_W)} \quad (1)$$

Finally, a simple word sense disambiguation method is applied, which identifies the most likely article in Wikipedia to which a concept should be linked to. The algorithm is based on statistical methods that identify the frequency of meanings in text, combined with symbolic methods that attempt to maximize the overlap between the current document and the candidate Wikipedia articles. See (Mihalcea and Csomai, 2007) for more details.

2.3 Biased Ranking of the Wikipedia Graph

Starting with the graph of encyclopedic knowledge, and knowing the nodes that belong to the input document, we want to rank all the nodes in the graph so that we obtain a score that indicates their importance relative to the given document. We can do this by using a graph-ranking algorithm *biased* toward the nodes belonging to the input document.

Graph-based ranking algorithms such as PageRank are essentially a way of deciding the importance of a vertex within a graph, based on global information recursively drawn from the entire graph. One formulation is in terms of a random walk through a directed graph. A "random surfer" visits nodes of the graph, and has some probability of jumping to some other random node of the graph. The rank of a node is an indication of the probability that one would find the surfer at that node at any given time.

Formally, let $G = (V, E)$ be a directed graph with the set of vertices V and set of edges E, where E is a subset of $V \times V$. For a given vertex V_i, let $In(V_i)$ be the set of vertices that point to it (predecessors), and let $Out(V_i)$ be the set of vertices that vertex V_i points to (successors). The PageRank score of a vertex V_i is defined as follows (Brin and Page, 1998):

$$S(V_i) = (1-d) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

where d is a damping factor usually set to 0.85.

In a "random surfer" interpretation of the ranking process, the $(1-d)$ portion represents the probability that a surfer navigating the graph will jump to a given node from any other node at random, and the summation portion indicates that the process will enter the node via edges directly connected to it. Using a method inspired by earlier work (Haveliwala, 2002), we modify the formula so that the $(1-d)$ component also accounts for the importance of the concepts found in the input document, and it is suppressed for all the nodes that are not found in the input document.

$$S(V_i) = (1-d) * Bias(V_i) + d * \sum_{j \in In(V_i)} \frac{1}{|Out(V_j)|} S(V_j)$$

where $Bias(V_i)$ is only defined for those nodes initially identified in the input document:

$$Bias(V_i) = \frac{f(V_i)}{\sum_{j \in InitalNodeSet} f(V_j)}$$

and 0 for all other nodes in the graph. $InitalNodeSet$ is the set of nodes belonging to the input document.

Note that $f(V_i)$ can vary in complexity from a default value of 1 to a complex knowledge-based estimation. In our implementation, we use a combination of the "keyphraseness" score assigned to the node V_i and its distance from the "Fundamental" category in Wikipedia.

3 Experiments

We run two experiments, aimed at measuring the relevancy of the automatically identified topics with respect to a manually annotated gold standard data set.

In the first experiment, the identification of the important concepts in the input text (used to bias the topic ranking process) is performed manually, by the Wikipedia users. In the second experiment, the identification of these important concepts is done automatically with the Wikify! system. In both experiments, the ranking of the concepts from the encyclopedic graph is performed using the dynamic ranking process described in Section 2.

We use a data set consisting of 150 articles from Wikipedia, which have been explicitly removed from the encyclopedic graph. All the articles in this data set include manual annotations of the relevant categories, as assigned by the Wikipedia users, against which we can measure the quality of the automatic topic assignments. The 150 articles have been randomly selected while following the constraint that they each contain at least three article links and at least three category links. Our task is to rediscover the relevant categories for each page. Note that the task is non-trivial, since there are more than 350,000 categories to choose from. We evaluate the quality of our system through the standard measures of precision and recall.

3.1 Manual Annotation of the Input Text

In this first experiment, the articles in the gold standard data set also include manual annotations of the important concepts in the text, i.e., the links to other Wikipedia articles as created by the Wikipedia users. Thus, in this experiment we only measure the accuracy of the dynamic topic ranking process, without interference from the Wikify! system.

There are two main parameters that can be set during a system run. First, the set of initial nodes used as bias in the ranking can include: (1) the initial set of articles linked to by the original document (via the Wikipedia links); (2) the categories listed in the articles linked to by the original document[2]; and (3) both. Second, the dynamic ranking process can be run through propagation on an encyclopedic graph that includes (1) all the articles from Wikipedia; (2) all the categories from Wikipedia; or (3) all the articles and the categories from Wikipedia.

Figures 1 and 2 show the precision and recall for the various settings. $Bias$ and $Propagate$ indicate the selections made for the two parameters, which can be set to either $Articles$, $Categories$, or $Both$.

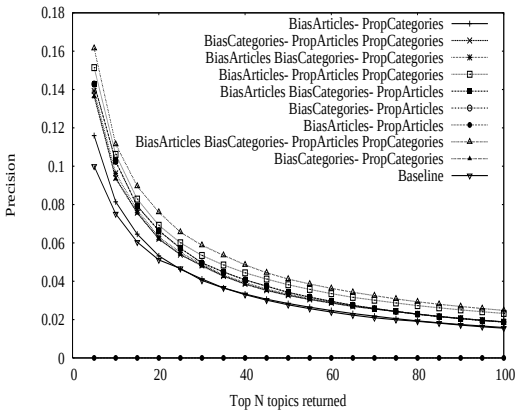


Figure 1: Precision for manual input text annotations.

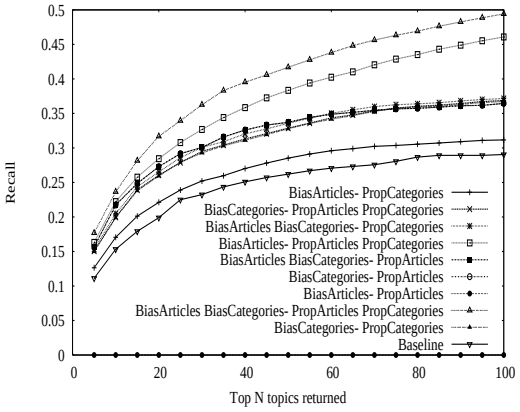


Figure 2: Recall for manual input text annotations.

As seen in the figures, the best results are obtained for a setting where both the initial bias and the propagation include all the available nodes, i.e., both articles and categories. Although the primary task is the identification of the categories, the addition of the article links improves the system performance.

[2]These should not be confused with the categories included in the document itself, which represent the gold standard annotations and are not used at any point.

To place results in perspective, we also calculate a baseline (labeled as "Baseline" in the plots), which selects by default all the categories listed in the articles linked to by the original document.

3.2 Automatic Annotation of the Input Text

The second experiment is similar to the first one, except that rather than using the manual annotations of the important concepts in the input document, we use instead the Wikify! system that automatically identifies these important concepts by using the method briefly described in Section 2.2. The article links identified by Wikify! are treated in the same way as the human anchor annotations from the previous experiment. In this experiment, we have an additional parameter, which consists of the percentage of links selected by Wikify! out of the total number of words in the document. We refer to this parameter as keyRatio. The higher the keyRatio, the more terms are added, but also the higher the potential of noise due to mis-disambiguation.

Figures 3 and 4 show the effect of varying the value of the keyRatio parameter on the precision and recall of the system. Note that in this experiment, we only use the best setting for the other two parameters as identified in the previous experiment, namely an initial bias and a propagation step that include all available nodes, i.e., both articles and categories.

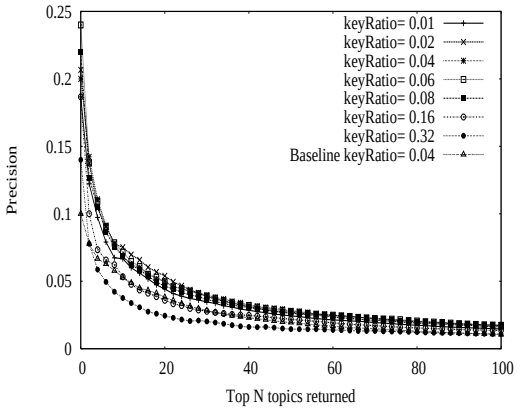


Figure 3: Precision for automatic input text annotations

The system's best performance occurs for a key ratio of 0.04 to 0.06, which coincides with the ratio found to be optimal in previous experiments using the Wikify! system (Mihalcea and Csomai, 2007).

Overall, the system manages to find many relevant topics for the documents in the evaluation data set, despite the large number of candidate topics (more than 350,000). Additional experiments performed against a set of documents from a source other than Wikipedia are reported in (Coursey et al., 2009).

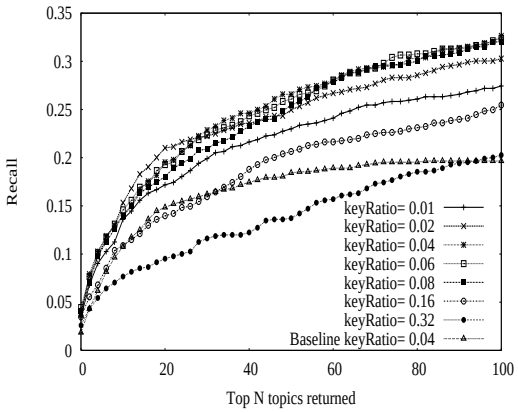


Figure 4: Recall for automatic input text annotations

4 Conclusions

In this paper, we presented an unsupervised system for automatic topic identification, which relies on a biased graph centrality algorithm applied on a graph built from Wikipedia. Our experiments demonstrate the usefulness of external encyclopedic knowledge for the task of topic identification.

Acknowledgments

This work has been partially supported by award #CR72105 from the Texas Higher Education Coordinating Board and by an award from Google Inc. The authors are grateful to the Waikato group for making their data set available.

References

S. Brin and L. Page. 1998. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks and ISDN Systems*, 30(1–7).

K. Coursey, R. Mihalcea, and W. Moen. 2009. Using encyclopedic knowledge for automatic topic identification. In *Proceedings of the Conference on Natural Language Learning*, Boulder, CO.

T. Haveliwala. 2002. Topic-sensitive PageRank. In *Proceedings of the Eleventh International World Wide Web Conference*, May.

O. Medelyan and I. H. Witten. 2008. Topic indexing with Wikipedia. In *Proceedings of the AAAI WikiAI workshop*.

R. Mihalcea and A. Csomai. 2007. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management*, Lisbon, Portugal.

Extracting Bilingual Dictionary from Comparable Corpora with Dependency Heterogeneity

Kun Yu **Junichi Tsujii**
Graduate School of Information Science and Technology
The University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo, Japan
{kunyu, tsujii}@is.s.u-tokyo.ac.jp

Abstract

This paper proposes an approach for bilingual dictionary extraction from comparable corpora. The proposed approach is based on the observation that a word and its translation share similar dependency relations. Experimental results using 250 randomly selected translation pairs prove that the proposed approach significantly outperforms the traditional context-based approach that uses bag-of-words around translation candidates.

1 Introduction

Bilingual dictionary plays an important role in many natural language processing tasks. For example, machine translation uses bilingual dictionary to reinforce word and phrase alignment (Och and Ney, 2003), cross-language information retrieval uses bilingual dictionary for query translation (Grefenstette, 1998). The direct way of bilingual dictionary acquisition is aligning translation candidates using parallel corpora (Wu, 1994). But for some languages, collecting parallel corpora is not easy. Therefore, many researchers paid attention to bilingual dictionary extraction from comparable corpora (Fung, 2000; Chiao and Zweigenbaum, 2002; Daille and Morin, 2008; Robitaille et al., 2006; Morin et al., 2007; Otero, 2008), in which texts are not exact translation of each other but share common features.

Context-based approach, which is based on the observation that a term and its translation appear in similar lexical contexts (Daille and Morin, 2008), is the most popular approach for extracting bilingual dictionary from comparable corpora and has shown its effectiveness in terminology extraction (Fung, 2000; Chiao and Zweigenbaum, 2002; Robitaille et al., 2006; Morin et al., 2007). But it only concerns about the lexical context around translation candidates in a restricted window. Besides, in comparable corpora, some words may appear in similar context even if they are not translation of each other. For example, using a Chinese-English comparable corpus from Wikipedia and following the definition in (Fung, 1995), we get context heterogeneity vector of three words (see Table 1). The Euclidean distance between the vector of ‘经济学(economics)’ and ‘economics’ is 0.084. But the Euclidean distance between the vector of ‘经济学’ and ‘medicine’ is 0.075. In such case, the incorrect dictionary entry ‘经济学/medicine’ will be extracted by context-based approach.

Table 1. Context heterogeneity vector of words.

Word	Context Heterogeneity Vector
经济学(economics)	(0.185, 0.006)
economics	(0.101, 0.013)
medicine	(0.113,0.028)

To solve this problem, we investigate a comparable corpora from Wikipedia and find the following phenomenon: *if we preprocessed the corpora with a dependency syntactic analyzer, a word in source language shares similar head and modifiers with its translation in target language, no matter whether they occur in similar context or not*. We call this phenomenon as ***dependency heterogeneity***. Based on this observation, we propose an approach to extract bilingual dictionary from comparable corpora. Not like only using bag-of-words around translation candidates in context-based approach, the proposed approach utilizes the syntactic analysis of comparable corpora to recognize the meaning of translation candidates. Besides, the lexical information used in the proposed approach does not restrict in a small window, but comes from the entire sentence.

We did experiments with 250 randomly selected translation pairs. Results show that compared with the approach based on context heterogeneity, the proposed approach improves the accuracy of dictionary extraction significantly.

2 Related Work

In previous work about dictionary extraction from comparable corpora, using context similarity is the most popular one.

At first, Fung (1995) utilized context heterogeneity for bilingual dictionary extraction. Our proposed approach borrows Fung’s idea but extends context heterogeneity to dependency heterogeneity, in order to utilize rich syntactic information other than bag-of-words.

After that, researchers extended context heterogeneity vector to context vector with the aid of an existing bilingual dictionary (Fung, 2000; Chiao and Zweigenbaum, 2002; Robitaille et al., 2006; Morin et al., 2007; Daille and Morin, 2008). In these works, dictionary extraction

Proceedings of NAACL HLT 2009: Short Papers, pages 121–124,
Boulder, Colorado, June 2009.

is fulfilled by comparing the similarity between the context vectors of words in target language and the context vectors of words in source language using an external dictionary. The main difference between these works and our approach is still our usage of syntactic dependency other than bag-of-words. In addition, except for a morphological analyzer and a dependency parser, our approach does not need other external resources, such as the external dictionary. Because of the well-developed morphological and syntactic analysis research in recent years, the requirement of analyzers will not bring too much burden to the proposed approach.

Besides of using window-based contexts, there were also some works utilizing syntactic information for bilingual dictionary extraction. Otero (2007) extracted lexico-syntactic templates from parallel corpora first, and then used them as seeds to calculate similarity between translation candidates. Otero (2008) defined syntactic rules to get lexico-syntactic contexts of words, and then used an external bilingual dictionary to fulfill similarity calculation between the lexico-syntactic context vectors of translation candidates. Our approach differs from these works in two ways: (1) both the above works defined syntactic rules or templates by hand to get syntactic information. Our approach uses data-driven syntactic analyzers for acquiring dependency relations automatically. Therefore, it is easier to adapt our approach to other language pairs. (2) the types of dependencies used for similarity calculation in our approach are different from Otero's work. Otero (2007; 2008) only considered about the modification dependency among nouns, prepositions and verbs, such as the adjective modifier of nouns and the object of verbs. But our approach not only uses modifiers of translation candidates, but also considers about their heads.

3 Dependency Heterogeneity of Words in Comparable Corpora

Dependency heterogeneity means *a word and its translation share similar modifiers and head in comparable corpora*. Namely, the modifiers and head of unrelated words are different even if they occur in similar context.

Table 2. Frequently used modifiers (words are not ranked).

经济学(economics)	**economics**	**medicine**
微观/micro	keynesian	physiology
宏观/macro	*new*	Chinese
计量/computation	institutional	traditional
新/new	positive	biology
政治/politics	*classical*	internal
大学/university	labor	science
古典派/classicists	*development*	clinical
发展/development	engineering	veterinary
理论/theory	finance	western
实证/demonstration	international	agriculture

For example, Table 2 collects the most frequently used 10 modifiers of the words listed in Table 1. It shows there are 3 similar modifiers (italic words) between '经济学(economics)' and 'economics'. But there is no similar word between the modifiers of '经济学' and that of 'medicine'. Table 3 lists the most frequently used 10 heads (when a candidate word acts as subject) of the three words. If excluding copula, '经济学' and 'economics' share one similar head (italic words). But '经济学' and 'medicine' shares no similar head.

Table 3. Frequently used heads
(the predicate of subject, words are not ranked).

经济学(economics)	**economics**	**medicine**
是/is	is	is
均衡/average	has	tends
毕业/graduate	was	include
承认/admit	*emphasizes*	moved
能/can	non-rivaled	means
分化/split	became	requires
剩下/leave	assume	includes
比/compare	relies	were
成为/become	can	has
偏重/emphasize	replaces	may

4 Bilingual Dictionary Extraction with Dependency Heterogeneity

Based on the observation of dependency heterogeneity in comparable corpora, we propose an approach to extract bilingual dictionary using dependency heterogeneity similarity.

4.1 Comparable Corpora Preprocessing

Before calculating dependency heterogeneity similarity, we need to preprocess the comparable corpora. In this work, we focus on Chinese-English bilingual dictionary extraction for single-nouns. Therefore, we first use a Chinese morphological analyzer (Nakagawa and Uchimoto, 2007) and an English pos-tagger (Tsuruoka et al., 2005) to analyze the raw corpora. Then we use MaltParser (Nivre et al., 2007) to get syntactic dependency of both the Chinese corpus and the English corpus. The dependency labels produced by MaltParser (e.g. SUB) are used to decide the type of heads and modifiers.

After that, the analyzed corpora are refined through following steps: (1) we use a stemmer[1] to do stemming for the English corpus. Considering that only nouns are treated as translation candidates, we use stems for translation candidate but keep the original form of their heads and modifiers in order to avoid excessive stemming. (2) stop words are removed. For English, we use the stop word list from (Fung, 1995). For Chinese, we remove '的(of)' as stop word. (3) we remove the dependencies including punctuations and remove the sentences with

[1] http://search.cpan.org/~snowhare/Lingua-Stem-0.83/

more than *k* (set as 30 empirically) words from both English corpus and Chinese corpus, in order to reduce the effect of parsing error on dictionary extraction.

4.2 Dependency Heterogeneity Vector Calculation

Equation 1 shows the definition of dependency heterogeneity vector of a word *W*. It includes four elements. Each element represents the heterogeneity of a dependency relation. 'NMOD' (noun modifier), 'SUB' (subject) and 'OBJ' (object) are the dependency labels produced by MaltParser.

$$(H_{NMODHead}, H_{SUBHead}, H_{OBJHead}, H_{NMODMod}) \quad (1)$$

$$H_{NMODHead}(W) = \frac{\text{number of different heads of W with NMOD label}}{\text{total number of heads of W with NMOD label}}$$

$$H_{SUBHead}(W) = \frac{\text{number of different heads of W with SUB label}}{\text{total number of heads of W with SUB label}}$$

$$H_{OBJHead}(W) = \frac{\text{number of different heads of W with OBJ label}}{\text{total number of heads of W with OBJ label}}$$

$$H_{NMODMod}(W) = \frac{\text{number of different modifiers of W with NMOD label}}{\text{total number of modifiers of W with NMOD label}}$$

4.3 Bilingual Dictionary Extraction

After calculating dependency heterogeneity vector of translation candidates, bilingual dictionary entries are extracted according to the distance between the vector of W_s in source language and the vector of W_t in target language. We use Euclidean distance (see equation 2) for distance computation. The smaller distance between the dependency heterogeneity vectors of W_s and W_t, the more likely they are translations of each other.

$$D_H(W_s, W_t) = \sqrt{D_{NMODHead}^2 + D_{SUBHead}^2 + D_{OBJHead}^2 + D_{NMODMod}^2} \quad (2)$$

$$D_{NMODHead} = H_{NMODHead}(W_s) - H_{NMODHead}(W_t)$$

$$D_{SUBHead} = H_{SUBHead}(W_s) - H_{SUBHead}(W_t)$$

$$D_{OBJHead} = H_{OBJHead}(W_s) - H_{OBJHead}(W_t)$$

$$D_{NMODMod} = H_{NMODMod}(W_s) - H_{NMODMod}(W_t)$$

For example, following above definitions, we get dependency heterogeneity vector of the words analyzed before (see Table 4). The distances between these vectors are D_H(经济学, economics) = 0.222, D_H(经济学, medicine) = 0.496. It is clear that the distance between the vector of '经济学(economics)' and 'economics' is much smaller than that between '经济学' and 'medicine'. Thus, the pair '经济学/economics' is extracted successfully.

Table 4. Dependency heterogeneity vector of words.

Word	Dependency Heterogeneity Vector
经济学(economics)	(0.398, 0.677, 0.733, 0.471)
economics	(0.466, 0.500, 0.625, 0.432)
medicine	(0.748, 0.524, 0.542, 0.220)

5 Results and Discussion

5.1 Experimental Setting

We collect Chinese and English pages from Wikipedia[2] with inter-language link and use them as comparable corpora. After corpora preprocessing, we get 1,132,492 English sentences and 665,789 Chinese sentences for dependency heterogeneity vector learning. To evaluate the proposed approach, we randomly select 250 Chinese/English single-noun pairs from the aligned titles of the collected pages as testing data, and divide them into 5 folders. *Accuracy* (see equation 3) and *MMR* (Voorhees, 1999) (see equation 4) are used as evaluation metrics. The average scores of both *accuracy* and *MMR* among 5 folders are also calculated.

$$Accuracy = \sum_{i=1}^{N} t_i \Big/ N \quad (3)$$

$$t_i = \begin{cases} 1, & \text{if there exists correct translation in top } n \text{ ranking} \\ 0, & \text{otherwise} \end{cases}$$

$$MMR = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{rank_i}, \quad rank_i = \begin{cases} r_i, & \text{if } r_i < n \\ 0, & \text{otherwise} \end{cases} \quad (4)$$

n means top n evaluation,
r_i means the rank of the correct translation in top n ranking
N means the total number of words for evaluation

5.2 Results of Bilingual Dictionary Extraction

Two approaches were evaluated in this experiment. One is the context heterogeneity approach proposed in (Fung, 1995) (*context* for short). The other is our proposed approach (*dependency* for short).

The average results of dictionary extraction are listed in Table 5. It shows both the average *accuracy* and average *MMR* of extracted dictionary entries were improved significantly (McNemar's test, $p<0.05$) by the proposed approach. Besides, the increase of top5 evaluation was much higher than that of top10 evaluation, which means the proposed approach has more potential to extract precise bilingual dictionary entries.

Table 5. Average results of dictionary extraction.

	context		*dependency*	
	ave.accu	*ave.MMR*	*ave.accu*	*ave.MMR*
Top5	0.132	0.064	0.208(↑57.58%)	0.104(↑62.50%)
Top10	0.296	0.086	0.380(↑28.38%)	0.128(↑48.84%)

5.3 Effect of Dependency Heterogeneity Vector Definition

In the proposed approach, a dependency heterogeneity vector is defined as the combination of head and modifier heterogeneities. To see the effects of different dependency heterogeneity on dictionary extraction, we evaluated the proposed approach with different vector definitions, which are

only-head: $(H_{NMODHead}, H_{SUBHead}, H_{OBJHead})$
only-mod: $(H_{NMODMod})$
only-NMOD: $(H_{NMODHead}, H_{NMODMod})$

Table 6. Average results with different vector definitions.

	Top5		Top10	
	ave.accu	*ave.MMR*	*ave.accu*	*ave.MMR*
context	0.132	0.064	0.296	0.086
dependency	**0.208**	**0.104**	**0.380**	**0.128**
only-mod	0.156	0.080	0.336	0.103
only-head	0.176	0.077	0.336	0.098
*only-NMOD*s	0.200	0.094	0.364	0.115

[2] http://download.wikimedia.org

The results are listed in Table 6. It shows with any types of vector definitions, the proposed approach outperformed the *context* approach. Besides, if comparing the results of *dependency*, *only-mod*, and *only-head*, a conclusion can be drawn that head dependency heterogeneities and modifier dependency heterogeneities gave similar contribution to the proposed approach. At last, the difference between the results of *dependency* and *only-NMOD* shows the head and modifier with NMOD label contributed more to the proposed approach.

5.4 Discussion

To do detailed analysis, we collect the dictionary entries that are not extracted by *context* approach but extracted by the proposed approach (*good* for short), and the entries that are extracted by *context* approach but not extracted by the proposed approach (*bad* for short) from top10 evaluation results with their occurrence time (see Table 7). If neglecting the entries '护照/passports' and '上海/shanghai', we found that the proposed approach tended to extract correct bilingual dictionary entries if both the two words occurred frequently in the comparable corpora, but failed if one of them seldom appeared.

Table 7. Good and bad dictionary entries.

Good		***Bad***	
Chinese	English	Chinese	English
犹太人/262	jew/122	十字架/53	crucifixion/19
速度/568	velocity/175	水族箱/6	aquarium/31
历史/2298	history/2376	混合物/47	mixture/179
组织/1775	organizations/2194	砖/17	brick/66
运动/1534	movement/1541	量化/23	quantification/31
护照/76	passports/80	上海/843	shanghai/1247

But there are two exceptions: (1) although '上海(shanghai)' and 'shanghai' appeared frequently, the proposed approach did not extract them correctly; (2) both '护照(passport)' and 'passports' occurred less than 100 times, but they were recognized successfully by the proposed approach. Analysis shows the cleanliness of the comparable corpora is the most possible reason. In the English corpus we used for evaluation, many words are incorrectly combined with 'shanghai' by '**br**' (i.e. line break), such as 'airport**br**shanghai'. These errors affected the correctness of dependency heterogeneity vector of 'shanghai' greatly. Compared with the dirty resource of 'shanghai', only base form and plural form of 'passport' occur in the English corpus. Therefore, the dependency heterogeneity vectors of '护照' and 'passports' were precise and result in the successful extraction of this dictionary entry. We will clean the corpora to solve this problem in our future work.

6 Conclusion and Future Work

This paper proposes an approach, which not uses the similarity of bag-of-words around translation candidates but considers about the similarity of syntactic dependencies, to extract bilingual dictionary from comparable corpora. Experimental results show that the proposed approach outperformed the context-based approach significantly. It not only validates the feasibility of the proposed approach, but also shows the effectiveness of applying syntactic analysis in real application.

There are several future works under consideration including corpora cleaning, extending the proposed approach from single-noun dictionary extraction to multi-words, and adapting the proposed approach to other language pairs. Besides, because the proposed approach is based on the syntactic analysis of sentences with no more than *k* words (see Section 4.1), the parsing accuracy and the setting of threshold *k* will affect the correctness of dependency heterogeneity vector learning. We will try other thresholds and syntactic parsers to see their effects on dictionary extraction in the future.

Acknowledgments

This research is sponsored by Microsoft Research Asia Web-scale Natural Language Processing Theme.

References

Y.Chiao and P.Zweigenbaum. 2002. Looking for Candidate Translational Equivalents in Specialized, Comparable Corpora. *Proceedings of LREC 2002*.

B.Daille and E.Morin. 2008. An Effective Compositional Model for Lexical Alignment. *Proceedings of IJCNLP-08*.

P.Fung. 1995. Compiling Bilingual Lexicon Entries from a Non-parallel English-Chinese Corpus. *Proceedings of the 3rd Annual Workshop on Very Large Corpora*. pp. 173-183.

P.Fung. 2000. A Statistical View on Bilingual Lexicon Extraction from Parallel Corpora to Non-parallel Corpora. *Parallel Text Processing: Alignment and Use of Translation Corpora*. Kluwer Academic Publishers.

G.Grefenstette. 1998. The Problem of Cross-language Information Retrieval. *Cross-language Information Retrieval*. Kluwer Academic Publishers.

E.Morin et al.. 2007. Bilingual Terminology Mining – Using Brain, not Brawn Comparable Corpora. *Proceedings of ACL 2007*. T.Nakagawa and K.Uchimoto. 2007. A Hybrid Approach to Word Segmentation and POS Tagging. *Proceedings of ACL 2007*.

J.Nivre et al.. 2007. MaltParser: A Language-independent System for Data-driven Dependency Parsing. *Natural Language Engineering*. 13(2): 95-135.

F.Och and H.Ney. 2003. A Systematic Comparison of Various Statistical Alignment Models. *Computational Linguistics*, 29(1): 19-51.

P.Otero. 2007. Learning Bilingual Lexicons from Comparable English and Spanish Corpora. *Proceedings of MT Summit XI*. pp. 191-198.

P.Otero. 2008. Evaluating Two Different Methods for the Task of Extracting Bilingual Lexicons from Comparable Corpora. *Proceedings of LREC 2008 Workshop on Comparable Corpora*. pp. 19-26.

X.Robitaille et al.. 2006. Compiling French Japanese Terminologies from the Web. *Proceedings of EACL 2006*.

Y.Tsuruoka et al.. 2005. Developing a Robust Part-of-speech Tagger for Biomedical Text. *Advances in Informatics – 10th Panhellenic Conference on Informationcs*. LNCS 3746. pp. 382-392.

E.M.Voorhees. 1999. The TREC-8 Question Answering Track Report. *Proceedings of the 8th Text Retrieval Conference*.

D.Wu. 1994. Learning an English-Chinese Lexicon from a Parallel Corpus. *Proceedings of the 1st Conference of the Association for Machine Translation in the Americas*.

Domain Adaptation with Artificial Data for Semantic Parsing of Speech

Lonneke van der Plas
Department of Linguistics
University of Geneva
Geneva, Switzerland

James Henderson
Department of Computer Science
University of Geneva
Geneva, Switzerland

Paola Merlo
Department of Linguistics
University of Geneva
Geneva, Switzerland

{Lonneke.vanderPlas,James.Henderson,Paola.Merlo}@unige.ch

Abstract

We adapt a semantic role parser to the domain of goal-directed speech by creating an artificial treebank from an existing text treebank. We use a three-component model that includes distributional models from both target and source domains. We show that we improve the parser's performance on utterances collected from human-machine dialogues by training on the artificially created data without loss of performance on the text treebank.

1 Introduction

As the quality of natural language parsing improves and the sophistication of natural language understanding applications increases, there are several domains where parsing, and especially semantic parsing, could be useful. This is particularly true in adaptive systems for spoken language understanding, where complex utterances need to be translated into shallow semantic representation, such as dialogue acts.

The domain on which we are working is goal-directed system-driven dialogues, where a system helps the user to fulfil a certain goal, e.g. booking a hotel room. Typically, users respond with short answers to questions posed by the system. For example *In the South* is an answer to the question *Where would you like the hotel to be?* Parsing helps identifying the components (*In the South* is a PP) and semantic roles identify the PP as a locative, yielding the following slot-value pair for the dialogue act: *area=South*. A PP such as *in time* is not identified as a locative, whereas keyword-spotting techniques as those currently used in dialogue systems may produce *area=South* and *area=time* indifferently.

Statistical syntactic and semantic parsers need treebanks. Current available data is lacking in one or more respects: Syntactic/semantic treebanks are developed on text, while treebanks of speech corpora are not semantically annotated (e.g. Switchboard). Moreover, the available human-human speech treebanks do not exhibit the same properties as the system-driven speech on which we are focusing, in particular in their proportion of non-sentential utterances (NSUs), utterances that are not full sentences. In a corpus study of a subset of the human-human dialogues in the BNC, Fernández (2006) found that only 9% of the total utterances are NSUs, whereas we find 44% in our system-driven data.

We illustrate a technique to adapt an existing semantic parser trained on merged Penn Treebank/PropBank data to goal-directed system-driven dialogue by artificial data generation. Our main contribution lies in the framework used to generate artificial data for domain adaptation. We mimic the distributions over parse structures in the target domain by combining the text treebank data and the artificially created NSUs, using a three-component model. The first component is a hand-crafted model of NSUs. The second component describes the distribution over full sentences and types of NSUs as found in a minimally annotated subset of the target domain. The third component describes the distribution over the internal parse structure of the generated data and is taken from the source domain.

Our approach differs from most approaches to domain adaptation, which require some training on fully annotated target data (Nivre et al., 2007), whereas we use minimally annotated target data only to help determine the distributions in the artificially created data. It also differs from previ-

Proceedings of NAACL HLT 2009: Short Papers, pages 125–128,
Boulder, Colorado, June 2009.

ous work in domain adaptation by Foster (2007), where similar proportions of ungrammatical and grammatical data are combined to train a parser on ungrammatical written text, and by Weilhammer et al. (2006), who use interpolation between two separately trained models, one on an artificial corpus of user utterances generated by a hand-coded domain-specific grammar and one on available corpora. Whereas much previous work on parsing speech has focused on speech repairs, e.g. Charniak and Johnson (2001), we focus on parsing NSUs.

2 The first component: a model of NSUs

To construct a model of NSUs we studied a subset of the data under consideration: TownInfo. This small corpus of transcribed spoken human-machine dialogues in the domain of hotel/restaurant/bar search is gathered using the TownInfo tourist information system (Lemon et al., 2006).

The NSUs we find in our data are mainly of the type answers, according to the classification given in Fernández (2006). More specifically, we find short answers, plain and repeated affirmative answers, plain and helpful rejections, but also greetings.

Current linguistic theory provides several approaches to dealing with NSUs (Merchant, 2004; Progovac et al., 2006; Fernández, 2006). Following the linguistic analysis of NSUs as non-sentential small clauses (Progovac et al., 2006) that do not have tense or agreement functional nodes, we make the assumption that they are phrasal projections. Therefore, we reason, we can create an artificial data set of NSUs by extracting phrasal projections from an annotated treebank.

In the example given in the introduction, we saw a PP fragment, but fragments can be NPs, APs, etc. We define different types of NSUs based on the root label of the phrasal projection and define rules that allow us to extract NSUs (partial parse trees) from the source corpus.[1] Because the target corpus also contains full sentences, we allow full sentences to be taken without modification from the source treebank.

[1]Not all of these rules are simple extractions of phrasal projections, as described in section 4.

3 The two distributional components

The distributional model consists of two components. By applying the extraction rules to the source corpus we build a large collection of both full sentences and NSUs. The distributions in this collection follow the distributions of trees in the source domain (first distributional component). We then sample from this collection to generate our artificial corpus following distributions from the target domain (second distributional component).

The probability of an artificial tree $P(f_i(c_j))$ generated with an extraction rule f_i applied to a constituent from the source corpus c_j is defined as

$$P(f_i(c_j)) = P(f_i)P(c_j|f_i) \approx P_t(f_i)P_s(c_j|f_i)$$

The first distributional component originates from the source domain. It is responsible for the internal structure of the NSUs and full sentences extracted. $P_s(c_j|f_i)$ is the probability of the constituent taken from the source treebank (c_j), given that the rule f_i is applicable to that constituent.

Sampling is done according to distributions of NSUs and full sentences found in the target corpus ($P_t(f_i)$). As explained in section 2, there are several types of NSUs found in the target domain. This second component describes the distributions of types of NSUs (or full sentences) found in the target domain. It determines, for example, the proportion of NP NSUs that will be added to the artificial corpus.

To determine the target distribution we classified 171 (approximately 5%) randomly selected utterances from the TownInfo data, that were used as a development set.[2] In Table 1 we can see that 15.2 % of the trees in the artificial corpus will be NP NSUs.[3]

4 Data generation

We constructed our artificial corpus from sections 2 to 21 of the Wall Street Journal (WSJ) section of the Penn Treebank corpus (Marcus et al., 1993)

[2]We discarded very short utterances (yes, no, and greetings) since they don't need parsing. We also do not consider incomplete NSUs resulting from interruptions or recording problems.

[3]Because NSUs can be interpreted only in context, the same NSU can correspond to several syntactic categories: *South* for example, can be an noun, an adverb, or an adjective. In case of ambiguity, we divided the score up for the several possible tags. This accounts for the fractional counts.

Category	# Occ.	Perc.	Category	# Occ.	Perc.
NP	19.0	15.2	RB	1.7	1.3
JJ	12.7	10.1	DT	1.0	0.8
PP	12.0	9.6	CD	1.0	0.8
NN	11.7	9.3	Total frag.	70.0	56.0
VP	11.0	8.8	Full sents	55.0	44.0

Table 1: Distribution of types of NSUs and full sentences in the TownInfo development set.

merged with PropBank labels (Palmer et al., 2005). We included all the sentences from this dataset in our artificial corpus, giving us 39,832 full sentences. In accordance with the target distribution we added 50,699 NSUs extracted from the same dataset. We sampled NSUs according to the distribution given in Table 1. After the extraction we added a root FRAG node to the extracted NSUs[4] and we capitalised the first letter of each NSU to form an utterance.

There are two additional pre-processing steps. First, for some types of NSUs maximal projections are added. For example, in the subset from the target source we saw many occurrences of nouns without determiners, such as *Hotel* or *Bar*. These types of NSUs would be missed if we just extracted NPs from the source data, since we assume that NSUs are maximal projections. Therefore, we extracted single nouns as well and we added the NP phrasal projections to these nouns in the constructed trees. Second, not all extracted NSUs can keep their semantic roles. Extracting part of the sentence often severs the semantic role from the predicate of which it was originally an argument. An exception to this are VP NSUs and prepositional phrases that are modifiers, such as locative PPs, which are not dependent on the verb. Hence, we removed the semantic roles from the generated NSUs except for VPs and modifiers.

5 Experiments

We trained three parsing models on both the original non-augmented merged Penn Treebank/Propbank corpus and the artificially generated augmented treebank including NSUs. We ran a contrastive experiment to examine the usefulness of the three-component model by training two versions of the augmented model: One with and one without the target component.[5]

These models were tested on two test sets: a small corpus of 150 transcribed utterances taken from the TownInfo corpus, annotated with gold syntactic and semantic annotation by two of the authors[6]: the TownInfo test set. The second test set is used to compare the performance of the parser on WSJ-style sentences and consists of section 23 of the merged Penn Treebank/Propbank corpus. We will refer to this test set as the non-augmented test set.

5.1 The statistical parser

The parsing model is the one proposed in Merlo and Musillo (2008), which extends the syntactic parser of Henderson (2003) and Titov and Henderson (2007) with annotations which identify semantic role labels, and has competitive performance. The parser uses a generative history-based probability model for a binarised left-corner derivation. The probabilities of derivation decisions are modelled using the neural network approximation (Henderson, 2003) to a type of dynamic Bayesian Network called an Incremental Sigmoid Belief Network (ISBN) (Titov and Henderson, 2007).

The ISBN models the derivation history with a vector of binary latent variables. These latent variables learn to represent features of the parse history which are useful for making the current and subsequent derivation decisions. Induction of these features is biased towards features which are local in the parse tree, but can find features which are passed arbitrarily far through the tree. This flexible mechanism for feature induction allows the model to adapt to the parsing of NSUs without requiring any design changes or feature engineering.

5.2 Results

In Table 2, we report labelled constituent recall, precision, and F-measure for the three trained parsers (rows) on the two test sets (columns).[7] These mea-

[4]The node FRAG exists in the Penn Treebank. Our annotation does not introduce new labels, but only changes their distribution.

[5]The model without the target distribution has a uniform distribution over full sentences and NSUs and within NSUs a uniform distribution over the 8 types.

[6]This test set was constructed separately and is completely different from the development set used to determine the distributions in the target data.

[7]Statistical significance is determined using a stratified shuffling method, using software available at `http://www.cis.`

Training	Testing					
	TownInfo			PTB nonaug		
	Rec	Prec	F	Rec	Prec	F
PTB nonaug	69.4	76.7	72.9	81.4	82.1	81.7
PTB aug(+t)	81.4	77.8	79.5	81.3	82.0	81.7
PTB aug(−t)	62.6	64.3	63.4	81.2	81.9	81.6

Table 2: Recall, precision, and F-measure for the two test sets, trained on non-augmented data and data augmented with and without the target distribution component.

sures include both syntactic labels and semantic role labels.

The results in the first two lines of the columns headed *TownInfo* indicate the performance on the real data to which we are trying to adapt our parser: spoken data from human-machine dialogues. The parser does much better when trained on the augmented data. The differences between training on newspaper text and newspaper texts augmented with artificially created data are statistically significant ($p < 0.001$) and particularly large for recall: almost 12%.

The columns headed *PTB nonaug* show that the performance on parsing WSJ texts is not hurt by training on data augmented with artificially created NSUs (first vs. second line). The difference in performance compared to training on the non-augmented data is not statistically significant.

The last two rows of the TownInfo data show the results of our contrastive experiment. It is clear that the three-component model and in particular our careful characterisation of the target distribution is indispensable. The F-measure drops from 79.5% to 63.4% when we disregard the target distribution.

6 Conclusions

We have shown how a three-component model that consists of a model of the phenomenon being studied and two distributional components, one from the source data and one from the target data, allows one to create data artificially for training a semantic parser. Specifically, analysis and minimal annotation of only a small subset of utterances from the target domain of spoken dialogue systems suffices to determine a model of NSUs as well as the necessary target distribution. Following this framework we were able to improve the performance of a statistical parser on goal-directed spoken data extracted from human-machine dialogues without degrading the performance on full sentences.

upenn.edu/~dbikel/software.html.

Acknowledgements

The research leading to these results has received funding from the EU FP7 programme (FP7/2007-2013) under grant agreement nr 216594 (CLASSIC project: www.classic-project.org).

References

E. Charniak and M. Johnson. 2001. Edit detection and parsing for transcribed speech. In *Procs. NAACL*.

R. Fernández. 2006. *Non-sentential utterances in dialogue: classification resolution and use.* Ph.D. thesis, University of London.

J. Foster. 2007. Treebanks gone bad: Parser evaluation and retraining using a treebank of ungrammatical sentences. *International Journal of Document Analysis and Recognition*, 10:1–16.

J. Henderson. 2003. Inducing history representations for broad-coverage statistical parsing. In *Procs. NAACL-HLT*.

O. Lemon, K. Georgila, J. Henderson, and M. Stuttle. 2006. An ISU dialogue system exhibiting reinforcement learning of dialogue policies: generic slot-filling in the TALK in-car system. In *Procs. EACL*.

M. Marcus, B. Santorini, and M.A. Marcinkiewicz. 1993. Building a large annotated corpus of English: the Penn Treebank. *Comp. Ling.*, 19:313–330.

J. Merchant. 2004. Fragments and ellipsis. *Linguistics and Philosophy*, 27:661–738.

P. Merlo and G. Musillo. 2008. Semantic parsing for high-precision semantic role labelling. In *Procs. CONLL*.

J. Nivre, J. Hall, S. Kübler, R. McDonald, J. Nilsson, S. Riedel, and D. Yuret. 2007. The CoNLL 2007 shared task on dependency parsing. In *Procs. EMNLP-CoNLL*.

M. Palmer, D. Gildea, and P. Kingsbury. 2005. The Proposition Bank: An annotated corpus of semantic roles. *Comp. Ling.*, 31:71–105.

L. Progovac, K. Paesani, E. Casielles, and E. Barton. 2006. *The Syntax of Nonsententials:Multidisciplinary Perspectives*. John Benjamins.

I Titov and J Henderson. 2007. Constituent parsing with Incremental Sigmoid Belief Networks. In *Procs. ACL*.

K. Weilhammer, M. Stuttle, and S. Young. 2006. Bootstrapping language models for dialogue systems. In *Procs. Conf. on Spoken Language Processing*.

Extending Pronunciation Lexicons via Non-phonemic Respellings

Lucian Galescu
Florida Institute for Human and Machine Cognition
40 S Alcaniz St., Pensacola FL 32502, USA
lgalescu@ihmc.us

Abstract

This paper describes work in progress towards using non-phonemic respellings as an additional source of information besides spelling in the process of extending pronunciation lexicons for speech recognition and text-to-speech systems. Preliminary experimental data indicates that the approach is likely to be successful. The major benefit of the approach is that it makes extending pronunciation lexicons accessible to average users.

1 Introduction

Speech recognition (SR) systems use pronunciation lexicons to map words into the phoneme-like units used for acoustic modeling. Text-to-speech (TTS) systems also make use of pronunciation lexicons, both internally and as "exception dictionaries" meant to override the systems' internal grapheme-to-phoneme (G2P) convertors. There are many situations where users might want to augment the pronunciation lexicons of SR and TTS systems, ranging from minor fixes, such as adding a few new words or alternate pronunciations for existing words, to significant development efforts, such as adapting a speech system to a specialized domain, or developing speech systems for new languages by bootstrapping from small amounts of data (Kominek *et al*., 2008).

Unfortunately, extending the pronunciation lexicon (PL) is not an easy task. Getting expert help is usually impractical, yet users have little or no support if they want to tackle the job themselves. Where available, the user has to either know how to transcribe a word's pronunciation into the application's underlying phone set, or, in rare cases, use pronunciation-by-orthography, whereby word pronunciations are respelled using other words (e.g., "*Thailand*" is pronounced like "*tie land*"). The former method requires a certain skill that is clearly beyond the capabilities of the average user; the latter is extremely limited in scope.

What is needed is a method that would make it easy for the users to specify pronunciations themselves, without requiring them to be or become expert phoneticians. In this paper we will argue – with backing from some preliminary experiments – that non-phonemic respelllings might be an accessible intermediate representation that will allow speech systems to learn pronunciations directly from user input faster and more accurately.

2 Extending pronunciation lexicons

Automatic G2P conversion seems the ideal tool to help users with PL expansion. The user would be shown a ranked list of automatically derived pronunciations and would have to pick the correct one. To make such a system more user-friendly, a synthesized waveform could also be presented (Davel and Barnard, 2004; also Kominek *et al*., 2008). This approach has a major drawback: if the system's choices are all wrong – which is, in fact, to be expected, if the number of choices is small – the user would have to provide their own pronunciation by using the system's phonetic alphabet. In our opinion this precludes the approach from being used by non-specialists.

Other systems try to learn pronunciations only from user-provided audio samples, via speech recognition/alignment (Beaufays *et al*., 2003; see also Bansal *et al*., 2009 and Chung *et al*., 2004). In such systems G2P conversion may be used to constrain choices, thereby overcoming the notoriously poor phone-level recognition performance. For example, Beaufays *et al*. (2003) focused on a directory assistance SR task, with many out-of-vocabulary proper names. Their procedure works by initializing a hypothesis by G2P conversion, and thereafter refining it with hypotheses from the joint alignment of phone lattices obtained from audio samples and the current best hypothesis. Several transformation rules were employed to expand the search space of alternative pronunciations.

While audio-based pronunciation learning may appear to be more user-friendly, it actually suffers from being a slow approach, with many audio samples being needed to achieve reasonable performance (the studies cited used up to 15 samples). It is also unclear whether the pronunciations learned are in fact correct, since the approach was mostly used to help increase the performance of a SR system. The SR performance improvements (ranging from 40% to 74%) must be due to better

Proceedings of NAACL HLT 2009: Short Papers, pages 129–132,
Boulder, Colorado, June 2009.

pronunciations, but we are not aware of the existence of any correctness evaluations.

3 Non-phonemic respellings

The method proposed here is aimed at allowing users to directly indicate the pronunciation of a word via *non-phonemic respellings* (NPRs). With NPRs, a word's pronunciation is represented according to the ordinary spelling rules of English, without attempting to represent each sound with a unique symbol. For example, the pronunciation of the word *phoneme* could be indicated as \FO-neem\, where capitalization indicates stress (boldface, underlining, and the apostrophe are also used as stress markers). It is often possible to come up with different respellings, and, indeed, systematicity is not a goal here; rather, the goal is to convey information about pronunciation using familiar spelling-to-sound rules, with no special training or tables of unfamiliar symbols.

NPRs are used to indicate the pronunciation of unfamiliar or difficult words by news organizations (mostly for foreign names), the United States Pharmacopoeia (for drug names), as well as countless interest groups (astronomy, horticulture, philosophy, etc.). Lately, Merriam-Webster Online[1] has started using NPRs in their popular Word of the Day[2] feature. Here is a recent example:

girandole • \JEER-un-dohl\

While NPRs seem to be used by a fairly wide range of audiences, we mustn't assume that most people are familiar with them. What we do know, however, is that people can learn new pronunciations faster and with fewer errors from NPRs than from phonemic transcriptions and this holds true whether they are linguistically-trained or not (Fraser, 1997). We contend, based on preliminary observations, that not only are NPRs easily decoded, but people seem to be able to produce relatively accurate NPRs, too.

4 Our Approach

Our vision is that speech applications would employ user-provided NPRs as an additional source of information besides orthography, and use dedicated NPR-to-pronunciation (N2P) models to derive hypotheses about the correct pronunciation.

However, before embarking on this project, we ought to answer three questions:

1. Is generic knowledge about grapheme-to-phoneme mappings in English sufficient to decode pronunciation respellings? Or, in technical terms, are generic G2P models going to work as N2P models?
2. Are pronunciation respellings useful in obtaining the correct pronunciation of a word beyond the capabilities of a G2P converter?
3. Since we don't require that average users learn a respelling system, are novice users able to generate useful respellings?

In the following we try to answer experimentally the technical counterparts of the first two questions, and report results of a small study designed to answer the third one.

4.1 Data and models

We collected a corpus of 2730 words with a total of 2847 NPR transcriptions (some words have multiple NPRs) from National Cancer Institute's Dictionary of Cancer Terms.[3] The dictionary contains over 4000 medical terms. Here are a couple of entries (without the definitions):

lactoferrin (LAK-toh-fayr-in)
valproic acid (val-PROH-ik A-sid)

Of the 2730 words, 1183 appear in the CMU dictionary (Weide, 1998) – we'll call this the ID set. Of note, about 180 words were not truly in-dictionary; for example, *Versed* (a drug brand name), pronounced \V ER0 S EH1 D\, is different from the in-dictionary word *versed*, pronounced \V ER1 S T\. We manually aligned all NPRs in the ID set with the phonetic transcriptions.

We transcribed phonetically another 928 of the words – we'll call this the OOD set – not found in the CMU dictionary; we verified the phonetic transcriptions against the Merriam-Webster Online Medical Dictionary and the New Oxford American Dictionary (McKean, 2005).

For G2P conversion we used a joint 4-gram model (Galescu, 2001) trained on automatic alignments for all entries in the CMU dictionary. We note that joint n-gram models seem to be among the best G2P models available (Polyakova and Bonafonte, 2006; Bisani and Ney, 2008).

4.2 Adequacy of generic G2P models

To answer the first question above, we looked at whether the generic joint 4-gram G2P model is adequate for converting NPRs into phonemes.

At first, it appeared that the answer would be negative. We found out that NPRs use GP correspondences that do not exist or are extremely rare in the CMU dictionary. For example, the <[*ih*], \IH\> correspondence is very infrequent in the

[1] http://www.merriam-webster.com
[2] http://www.merriam-webster.com/cgi-bin/mwwod.pl
[3] http://www.cancer.gov

CMU dictionary (and appears only in proper names, e.g., *Stihl*), but is very frequently used in NPRs. Therefore, for the [*ih*] grapheme the G2P converter prefers \IH HH\ to the intended \IH\. Similar problems happen because of the way some diphones are transcribed. Two other peculiarities of the transcription accounted for other errors: a) always preferring /S/ in plurals where /Z/ would be required, and b) using [*ayr*] to transcribe \EH R\, which uses the very rare <[*ay*], \EH\> mapping. These deviations from ordinary GP correspondences occur with regularity and therefore we were able to fix them with four post-processing rules. We are confident that these rules capture specific choices made during the compilation of the Dictionary of Cancer Terms, to reduce ambiguity, and increase consistency, with the expectation that readers would learn to make the correct phonological choices when reading the respellings.

Another issue was that the set of GP mappings used in NPRs was extremely small (111) compared to the GP correspondence set obtained automatically from the CMU dictionary (1130, many of them occurring only in proper names). However, it turns out that 47524 entries in the CMU dictionary (about 45%) use exclusively GP mappings found in NPRs! This suggests that, while the generic G2P model may not be adequate for the N2P task, the GP mappings used in NPRs are sufficiently common that a more adequate N2P model could be built from generic dictionary entries by selecting only relevant entries for training. Unfortunately we don't have a full account of all "exotic" entries in the CMU dictionary, but we expect that by simply removing from the training data the approximately 54K known proper names will yield a reasonable starting point for building N2P models.

4.3 NPR-to-pronunciation conversion

To assess the contribution of NPR information to pronunciation prediction, we compare the performance of spelling-to-pronunciation conversion (the baseline) to that of NPR-to-pronunciation conversion, as well as to that of a combined spelling and NPR-based conversion, which is our end goal.

For the N2P task, we trained two joint 4-gram models: one based on the aligned NPRs, and a second based on the 47K CMU dictionary entries that use only GP mappings found in NPRs. Then, we interpolated the two models to obtain an NPR-specific model (the weights were not optimized for these experiments), which we'll call the N2P model. The combined, spelling and NPR-based model was an oracle combination of the G2P and the N2P model. Phone error rates (PER) and word error rates (WER) for both the ID set and the OOD set are shown in Figures 1 and 2, respectively. We

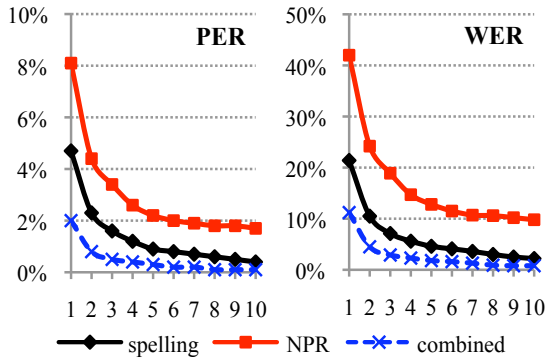


Figure 1. Phone and word error rates on the ID set.

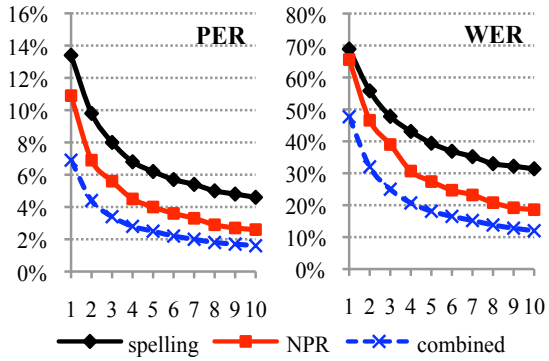


Figure 2. Phone and word error rates for the OOD set.

obtained n-best pronunciations with n from 1 to 10 for the three models considered.

As expected, G2P performance is very good on the ID set, since the test data was used in training the G2P model. Significantly, even though the N2P model is not as good itself, the combined model shows marked error rate reductions: for the top hypothesis it cuts the PER by over 57%, and the WER by over 47% when compared to the G2P performance on spelling alone.

Since the OOD set represents data unseen by either the spelling-based model or the NPR-based model, all models' performance is severely degraded compared to that on the ID set. But here we see that NPR-based pronunciations are already better than spelling-based ones. For the top hypothesis, compared to the performance of the G2P model alone, the N2P model shows almost 19% better PER, and almost 5% better WER, whereas the combined model achieves 49% better PER and close to 31% better WER.

4.4 User-generated NPRs

To answer the third question, we collected user-generated NPRs from five subjects. The subjects were all computer-savvy, with at least a BSc degree. Only one subject expressed some familiarity with NPRs (but didn't generate better NPRs than

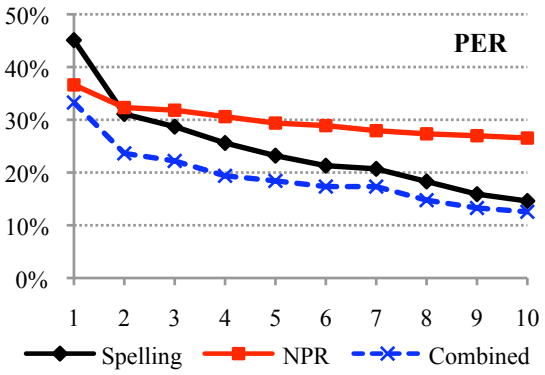


Figure 3. Phone error rates for user-generated NPRs.

other subjects).

The subjects were shown four examples of NPRs; two of them were recent Word of the Day entries, and had audio attached to them. The other two were selected from the OOD set. With only four words and two different sources we wanted to ensure that users would not be able to train themselves to a specific system. Subjects understood the problem easily and rarely if ever looked back at the examples during the actual test.

The test involved generating NPRs for 20 of the most difficult words for our generic GP model from the OOD set (e.g., *bronchoscope*, *parenchyma*, etc.). These words turned out to be mostly unfamiliar to users as well (the average familiarity score was just under 1.9 on a 4-point scale. No audio and no feedback were given.

Users varied greatly in the choices they made. For the word *acupressure*, the first two syllables were transcribed as AK-YOO in the Dictionary of Cancer Terms, and users came up with ACK-YOU, AK-U, and AK-YOU. This underscores that a good N2P model would have to account for far more GP mappings than the 111 found in our data.

Sometimes users had trouble assigning consonants to syllables (syllabification wasn't required, but subjects tried anyway), on occasion splitting them across syllable boundaries (e.g., \BIL-LIH-RUE-BEN\ for *bilirubin*), which guarantees an insertion error. It is quite likely that some error model might be required to deal with such issues.

Nonetheless, even though imperfect, the resulting NPRs showed excellent promise. Looking just at the top hypothesis, whereas the average PER on those 20 words was about 45% for the G2P model, pronunciations obtained from NPRs using the same G2P model (new GP mappings precluded the use of the N2P model described in the previous section) had only around 36% (+/-5%) phone error rate. The combined model showed an even better performance of about 33% (+/-5%) PER. Full results for n-best lists up to n=10 are shown in Figure 3.

5 Conclusions and Further Work

The experiments we conducted are preliminary, and most of the work remains to be done. More data need to be collected and analyzed before good NPR-to-pronunciation models can be trained. Further investigations need to be conducted to assess the average users' ability to generate NPRs and how they tend to deviate from the general grapheme-to-phoneme rules of English.

Nonetheless, we believe these experiments give strong indications that NPRs would be an excellent source of information to improve the quality of pronunciation hypotheses generated from spelling. Moreover, it appears that novice users don't have much difficulty generating useful NPRs on their own; we expect that their skill would increase with use. Particularly useful would be for the system to be able to provide feedback, including generating NPRs; we have started investigating this reverse problem, of obtaining NPRs from pronunciations, and are encouraged by the initial results.

References

D. Bansal, N. Nair, R. Singh, and B. Raj. 2009. A Joint Decoding Algorithm for Multiple-Example-Based Addition of Words to a Pronunciation Lexicon. *Proc. ICASSP'2009*, pp. 2104-2107.

F. Beaufays, et al. 2003. Learning Linguistically Valid Pronunciation From Acoustic Data. *Proc. Eurospeech'03*, Geneva, pp. 2593-2596.

M. Bisani and H. Ney. 2008. Joint-Sequence Models for Grapheme-to-Phoneme Conversion. *Speech Communication*, 50(5):434-451.

G. Chung, C. Wang, S. Seneff, E. Filisko, and M. Tang. 2004. Combining Linguistic Knowledge and Acoustic Information in Automatic Pronunciation Lexicon Generation. *Proc. Interspeech'04*, Jeju Island, Korea.

M. Davel and E. Barnard. 2004. The Efficient Generation of Pronunciation Dictionaries: Human Factors during Bootstrapping. *Proc. INTERSPEECH 2004*, Korea.

H. Fraser. 1997. Dictionary pronunciation guides for English. *International Journal of Lexicography*, 10(3), 181-208.

L. Galescu and J. Allen. 2001. Bi-directional Conversion Between Graphemes and Phonemes Using a Joint N-gram Model. *Proc. 4th ISCA Tutorial and Research Workshop on Speech Synthesis*, Scotland.

J. Kominek, S. Badaskar, T. Schultz, and A. Black. 2008. Improving Speech Systems Built from Very Little Data. *Proc. INTERSPEECH 2008*, Australia.

E. McKean (ed.). 2005. *The New Oxford American Dictionary* (2nd ed.). Oxford University Press.

T. Polyakova and A. Bonafonte, 2006. Learning from Errors in Grapheme-to-Phoneme Conversion. *Proc. ISCLP'2006*. Pittsburgh, USA.

R.L. Weide. 1998. The CMU pronunciation dictionary, release 0.6. http://www.speech.cs.cmu.edu/cgi-bin/cmudict.

A Speech Understanding Framework that Uses Multiple Language Models and Multiple Understanding Models

†Masaki Katsumaru, ‡Mikio Nakano, †Kazunori Komatani,
‡Kotaro Funakoshi, †Tetsuya Ogata, †Hiroshi G. Okuno

†Graduate School of Informatics, Kyoto University
Yoshida-Hommachi, Sakyo, Kyoto
606-8501, Japan
{katumaru, komatani}@kuis.kyoto-u.ac.jp
{ogata, okuno}@kuis.kyoto-u.ac.jp

‡Honda Research Institute Japan Co., Ltd.
8-1 Honcho, Wako, Saitama
351-0188, Japan
{nakano, funakoshi}@jp.honda-ri.com

Abstract

The optimal combination of language model (LM) and language understanding model (LUM) varies depending on available training data and utterances to be handled. Usually, a lot of effort and time are needed to find the optimal combination. Instead, we have designed and developed a new framework that uses multiple LMs and LUMs to improve speech understanding accuracy under various situations. As one implementation of the framework, we have developed a method for selecting the most appropriate speech understanding result from several candidates. We use two LMs and three LUMs, and thus obtain six combinations of them. We empirically show that our method improves speech understanding accuracy. The performance of the oracle selection suggests further potential improvements in our system.

1 Introduction

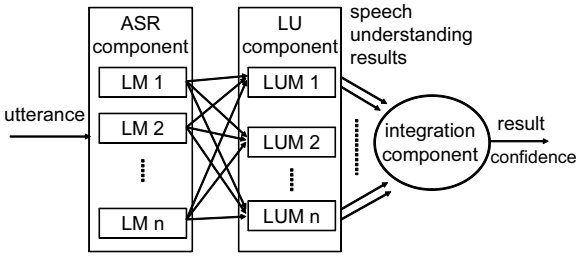
The speech understanding component in a spoken dialogue system consists of an automatic speech recognition (ASR) component and a language understanding (LU) component. To develop a speech understanding component, we need to prepare an ASR language model (LM) and a language understanding model (LUM) for the dialogue domain of the system. There are many types of LMs such as finite-state grammars and N-grams, and many types of LUMs such as finite-state transducers (FST), weighted finite-state transducers (WFST), and keyphrase-extractors (extractor). Selecting a suitable combination of LM and LUM is necessary for robust speech understanding against various user utterances.

Conventional studies of speech understanding have investigated which LM and LUM give the best performance by using fixed training and test data such as the Air Travel Information System (ATIS) corpus. However, in real system development, resources such as training data for statistical models and efforts to write finite-state grammars vary according to the available human resources or budgets. Domain-dependent training data are particularly difficult to obtain. Therefore, in conventional system development, system developers determine the types of LM and LUM by trial and error. Every LM and LUM has some advantages and disadvantages, so it is difficult for a single combination of LM and LUM to gain high accuracy except in a situation involving a lot of training data and effort. Therefore, using multiple speech understanding methods is a more effective approach.

In this paper, we propose a speech understanding framework called "Multiple Language models and Multiple Understanding models (MLMU)", in which multiple LMs and LUMs are used, to achieve better performance under the various development situations. It selects the best speech understanding result from the multiple results generated by arbitrary combinations of LMs and LUMs.

So far there have been several attempts to improve ASR and speech understanding using multiple speech recognizers and speech understanding modules. ROVER (Fiscus, 1997) tried to improve ASR accuracy by integrating the outputs of multiple ASRs with different acoustic and language mod-

Proceedings of NAACL HLT 2009: Short Papers, pages 133–136,
Boulder, Colorado, June 2009.



LM: Language Model
LUM: Language Understanding Model

Figure 1: Flow of speech understanding in MLMU

els. The work is different from our study in the following two points: it does not deal with speech understanding, and it assumes that each ASR is well-developed and achieves high accuracy for a variety of speech inputs. Eckert et al. (1996) used multiple LMs to deal with both in-grammar utterances and out-of-grammar utterances, but did not mention language understanding. Hahn et al. (2008) used multiple LUMs, but just a single language model.

2 Speech Understanding Framework MLMU

MLMU is a framework by which system developers can use multiple speech understanding methods by preparing multiple LMs and multiple LUMs. Figure 1 illustrates the flow of speech understanding in MLMU. System developers list available LMs and LUMs for each system's domain, and the system understands utterances by using these models. The framework selects one understanding result from multiple results or calculates a confidence score of the result by using the generated multiple understanding results.

MLMU can improve speech understanding for the following reason. The performance of each speech understanding (a combination of LM and LUM) might not be very high when either training data for the statistical model or available expertise and effort for writing grammar are insufficient. In such cases, some utterances might not be covered by the system's finite-state grammar LM, and probability estimation in the statistical models may not be very good. Using multiple speech understanding models is expected to solve this problem because each model has different specialities. For example, finite-state grammar LMs and FST-based LUMs achieve high accuracy in recognizing and understanding in-grammar utterances, whereas out-of-grammar utterances are covered by N-gram models and LUMs based on WFST and keyphrase-extractors. Therefore it is more possible that the understanding results of MLMU will include the correct result than a case when a single understanding model is used.

The understanding results of MLMU will be helpful in many ways. We used them to achieve better understanding accuracy by selecting the most reliable one. This selection is based on features concerning ASR results and language understanding results. It is also possible to delay the selection, holding multiple understanding result candidates that will be disambiguated as the dialogue proceeds (Bohus, 2004). Furthermore, confidence scores, which enable an efficient dialogue management (Komatani and Kawahara, 2000), can be calculated by ranking these results or by voting on them, by using multiple speech understanding results. The understanding results can be used in the discourse understanding module and the dialogue management module. They can choose one of the understanding results depending on the dialogue situation.

3 Implementation

3.1 Available Language Models and Language Understanding Models

We implemented MLMU as a library of RIME-TK, which is a toolkit for building multi-domain spoken dialogue systems (Nakano et al., 2008). With the current implementation, developers can use the following LMs:

1. A LM based on finite-state grammar (FSG)
2. A domain-dependent statistical N-gram model (N-gram)

and the following LUMs:

1. Finite-state transducer (FST)
2. Weighted FST (WFST)
3. Keyphrase-extractor (extractor).

System developers can use multiple finite-state-grammar-based LMs or N-gram-based LMs, and

also multiple FSTs and WFSTs. They can specify the combination for each domain by preparing LMs and LUMs. They can specify grammar models when sufficient human labor is available for writing grammar, and specify statistical models when a corpus for training models is available.

3.2 Selecting Understanding Result based on ASR and LU Features

We also implemented a mechanism for selecting one of the understanding results as the best hypothesis. The mechanism chooses the result with the highest estimated probability of correctness. Probabilities are estimated for each understanding result by using logistic regression, which uses several ASR and LU features.

We define P_i as the probability that speech understanding result *i* is correct, and we select one result based on $\underset{i}{\operatorname{argmax}} P_i$. We denote each speech understanding result as *i* (*i* = 1,...,6). We constructed a logistic regression model for P_i. The regression function can be written as:

$$P_i = \frac{1}{1 + exp(-(a_{i1}F_{i1} + \ldots + a_{im}F_{im} + b_i))}. \quad (1)$$

The coefficients $a_{i1}, \ldots, a_{im}, b_i$ were fitted using training data. The independent variables $F_{i1}, F_{i2}, ..., F_{im}$ are listed in Table 1. In the table, *n* indicates the number of understanding results, that is, *n* = 6 in this paper's experiment. Here, we denote the features as $F_{i1}, F_{i2}, ..., F_{im}$.

Features from F_{i1} to F_{i3} represent characteristics of ASR results. The acoustic scores were normalized by utterance durations in seconds. These features are used for verifying its ASR result. Features from F_{i4} to F_{i9} represent characteristics of LU results. Features from F_{i4} to F_{i6} are defined on the basis of the concept-based confidence scores (Komatani and Kawahara, 2000).

4 Preliminary Experiment

We conducted a preliminary experiment to show the potential of the framework by using the two LMs and three LUMs noted in Section 3.1.

Table 1: Features from speech understanding result *i*

F_{i1}:	acoustic score of ASR
F_{i2}:	difference between F_{i1} and acoustic score of ASR for utterance verification
F_{i3}:	utterance duration [sec.]
F_{i4}:	average confidence scores for concepts in *i*
F_{i5}:	average of F_{i4} ($\frac{1}{n}\sum_i^n F_{i4}$)
F_{i6}:	proportion of F_{i4} (F_{i4} / $\sum_i^n F_{i5}$)
F_{i7}:	average # concepts ($\frac{1}{n}\sum_i^n \#\mathrm{concept_i}$)
F_{i8}:	max. # concepts (max ($\#\mathrm{concept_i}$))
F_{i9}:	min. # concepts (min ($\#\mathrm{concept_i}$))

4.1 Preparing LMs and LUMs

The finite-state grammar rules were written in sentence units manually. A domain-dependent statistical N-gram model was trained on 10,000 sentences randomly generated from the grammar. The vocabulary sizes of the grammar LM and the domain-dependent statistical LM were both 278. We also used a domain-independent statistical N-gram model for obtaining acoustic scores for utterance verification, which was trained on Web texts (Kawahara et al., 2004). Its vocabulary size was 60,250.

The grammar used in the FST was the same as the FSG used as one of the LMs, which was manually written by a system developer. The WFST-based LU was based on a method to estimate WFST parameters with a small amount of data (Fukubayashi et al., 2008). Its parameters were estimated by using 105 utterances of just one user. The keyphrase extractor extracts as many concepts as possible from an ASR result on the basis of a grammar while ignoring words that do not match the grammar.

4.2 Target Data for Evaluation

We used 3,055 utterances in the rent-a-car reservation domain (Nakano et al., 2007). We used Julius (ver. 4.0.2) as the speech recognizer and a 3000-state phonetic tied-mixture (PTM) triphone model as the acoustic model[1]. ASR accuracy in mora accuracy when using the FSG and the N-gram model were 71.9% and 75.5% respectively. We used concept error rates (CERs) to represent the speech understanding accuracy, which is calculated as fol-

[1] http://julius.sourceforge.jp/

Table 2: CERs [%] for each speech understanding method

speech understanding method (LM + LUM)	CER
(1) FSG + FST	26.9
(2) FSG + WFST	29.9
(3) FSG + extractor	27.1
(4) N-gram + FST	35.2
(5) N-gram + WFST	25.3
(6) N-gram + extractor	26.0
selection from (1) through (6) (our method)	22.7
oracle selection	13.5

lows:

$$CER = \frac{\#\text{ error concepts}}{\#\text{concepts in utterances}}. \qquad (2)$$

We manually annotated whether an understanding result of each utterance was correct or not, and used them as training data to fit the coefficients $a_{i1}, \ldots, a_{im}, b_i$.

4.3 Evaluation in Concept Error Rates

We fitted the coefficients of regression functions and selected understanding results with a 10-fold cross validation. Table 2 lists the CERs based on combinations of single LM and LUM and by our method. Of all combinations of single LM and LUM, the best accuracy was obtained with (5) (N-gram + WFST). Our method improved by 2.6 points over (5). Although we achieved a lower CER, we used a lot of data to estimate logistic regression coefficients. Such a large amount of data may not be available in a real situation. We will conduct more experiments by changing the amount of training data. Table 2 also shows the accuracy of the oracle selection, which selected the best speech understanding result manually. The CER of the oracle selection was 13.5%, a significant improvement compared to all combinations of a LM and LUM. There is no combination of a LM and LUM whose understanding results were not selected at all in the oracle selection and our method's selection. These results show that using multiple LMs and multiple LUMs can potentially improve speech understanding accuracy.

5 Ongoing work

We will conduct more experiments in other domains or with other resources to evaluate the effectiveness of our framework. We plan to investigate the case in which a smaller amount of the training data is used to estimate the coefficients of the logistic regressions. Furthermore, finding a way to calculate confidence scores of speech understanding results is on our agenda.

References

Dan Bohus. 2004. *Error awareness and recovery in task-oriented spoken dialogue systems*. Ph.D. thesis, Carnegie Mellon University.

Wieland Eckert, Florian Gallwitz, and Heinrich Niemann. 1996. Combining stochastic and linguistic language models for recognition of spontaneous speech. In *Proc. ICASSP*, pages 423–426.

Jonathan G. Fiscus. 1997. A post-processing system to yield reduced word error rates: Recognizer Output Voting Error Reduction (ROVER). In *Proc. ASRU*, pages 347–354.

Yuichiro Fukubayashi, Kazunori Komatani, Mikio Nakano, Kotaro Funakoshi, Hiroshi Tsujino, Tetsuya Ogata, and Hiroshi G. Okuno. 2008. Rapid prototyping of robust language understanding modules for spoken dialogue systems. In *Proc. IJCNLP*, pages 210–216.

Stefan Hahn, Patrick Lehnen, and Hermann Ney. 2008. System combination for spoken language understanding. In *Proc. Interspeech*, pages 236–239.

Tatsuya Kawahara, Akinobu Lee, Kazuya Takeda, Katsunobu Itou, and Kiyohiro Shikano. 2004. Recent progress of open-source LVCSR Engine Julius and Japanese model repository. In *Proc. ICSLP*, pages 3069–3072.

Kazunori Komatani and Tatsuya Kawahara. 2000. Flexible mixed-initiative dialogue management using concept-level confidence measures of speech recognizer output. In *Proc. COLING*, volume 1, pages 467–473.

Mikio Nakano, Yuka Nagano, Kotaro Funakoshi, Toshihiko Ito, Kenji Araki, Yuji Hasegawa, and Hiroshi Tsujino. 2007. Analysis of user reactions to turn-taking failures in spoken dialogue systems. In *Proc. SIGdial*, pages 120–123.

Mikio Nakano, Kotaro Funakoshi, Yuji Hasegawa, and Hiroshi Tsujino. 2008. A framework for building conversational agents based on a multi-expert model. In *Proc. SIGdial*, pages 88–91.

Taking into Account the Differences between Actively and Passively Acquired Data: The Case of Active Learning with Support Vector Machines for Imbalanced Datasets

Michael Bloodgood*
Human Language Technology
Center of Excellence
Johns Hopkins University
Baltimore, MD 21211 USA
bloodgood@jhu.edu

K. Vijay-Shanker
Computer and Information
Sciences Department
University of Delaware
Newark, DE 19716 USA
vijay@cis.udel.edu

Abstract

Actively sampled data can have very different characteristics than passively sampled data. Therefore, it's promising to investigate using different inference procedures during AL than are used during passive learning (PL). This general idea is explored in detail for the focused case of AL with cost-weighted SVMs for imbalanced data, a situation that arises for many HLT tasks. The key idea behind the proposed InitPA method for addressing imbalance is to base cost models during AL on an estimate of overall corpus imbalance computed via a small unbiased sample rather than the imbalance in the labeled training data, which is the leading method used during PL.

1 Introduction

Recently there has been considerable interest in using active learning (AL) to reduce HLT annotation burdens. Actively sampled data can have different characteristics than passively sampled data and therefore, this paper proposes modifying algorithms used to infer models during AL. Since most AL research assumes the same learning algorithms will be used during AL as during passive learning[1] (PL), this paper opens up a new thread of AL research that accounts for the differences between passively and actively sampled data.

The specific case focused on in this paper is that of AL with SVMs (AL-SVM) for imbalanced datasets[2]. Collectively, the factors: interest in AL, widespread class imbalance for many HLT tasks, interest in using SVMs, and PL research showing that SVM performance can be improved substantially by addressing imbalance, indicate the importance of the case of AL with SVMs with imbalanced data.

Extensive PL research has shown that learning algorithms' performance degrades for imbalanced datasets and techniques have been developed that prevent this degradation. However, to date, relatively little work has addressed imbalance during AL (see Section 2). In contrast to previous work, this paper advocates that the AL scenario brings out the need to modify PL approaches to dealing with imbalance. In particular, a new method is developed for cost-weighted SVMs that estimates a cost model based on overall corpus imbalance rather than the imbalance in the so far labeled training data. Section 2 discusses related work, Section 3 discusses the experimental setup, Section 4 presents the new method called InitPA, Section 5 evaluates InitPA, and Section 6 concludes.

2 Related Work

A problem with imbalanced data is that the class boundary (hyperplane) learned by SVMs can be too close to the positive (pos) examples and then recall suffers. Many approaches have been presented for overcoming this problem *in the PL setting.* Many require substantially longer training times or ex-

*This research was conducted while the first author was a PhD student at the University of Delaware.

[1]Passive learning refers to the typical supervised learning setup where the learner does not actively select its training data.

[2]This paper focuses on the fundamental case of binary classification where class imbalance arises because the positive examples are rarer than the negative examples, a situation that naturally arises for many HLT tasks.

Proceedings of NAACL HLT 2009: Short Papers, pages 137–140,
Boulder, Colorado, June 2009.

tra training data to tune parameters and thus are not ideal for use during AL. Cost-weighted SVMs (cwSVMs), on the other hand, *are* a promising approach for use with AL: they impose no extra training overhead. cwSVMs introduce unequal cost factors so the optimization problem solved becomes:

Minimize:

$$\frac{1}{2}\|\vec{w}\|^2 + C_+ \sum_{i:y_i=+1} \xi_i + C_- \sum_{i:y_i=-1} \xi_i \quad (1)$$

Subject to:

$$\forall k : y_k \left[\vec{w} \cdot \vec{x}_k + b\right] \geq 1 - \xi_k, \quad (2)$$

where $(\vec{w}, b)$ represents the learned hyperplane, $\vec{x}_k$ is the feature vector for example k, y_k is the label for example k, $\xi_k = max(0, 1 - y_k(\vec{w}_k \cdot \vec{x}_k + b))$ is the slack variable for example k, and C_+ and C_- are user-defined cost factors.

The most important part for this paper are the cost factors C_+ and C_-. The ratio $\frac{C_+}{C_-}$ quantifies the importance of reducing slack error on pos train examples relative to reducing slack error on negative (neg) train examples. The value of the ratio is crucial for balancing the precision recall tradeoff well. (Morik et al., 1999) showed that during PL, setting $\frac{C_+}{C_-} = \frac{\text{\# of neg training examples}}{\text{\# of pos training examples}}$ is an effective heuristic. Section 4 explores using this heuristic *during AL* and explains a modified heuristic that could work better during AL.

(Ertekin et al., 2007) propose using the balancing of training data that occurs as a result of AL-SVM to handle imbalance and do not use any further measures to address imbalance. (Zhu and Hovy, 2007) used resampling to address imbalance and based the amount of resampling, which is the analog of our cost model, on the amount of imbalance in the current set of labeled train data, as PL approaches do. In contrast, the InitPA approach in Section 4 bases its cost models on overall (unlabeled) corpus imbalance rather than the amount of imbalance in the current set of labeled data.

3 Experimental Setup

We use relation extraction (RE) and text classification (TC) datasets and SVM^{light} (Joachims, 1999) for training the SVMs. For RE, we use AImed, previously used to train protein interaction extraction systems ((Giuliano et al., 2006)). As in previous work, we cast RE as a binary classification task

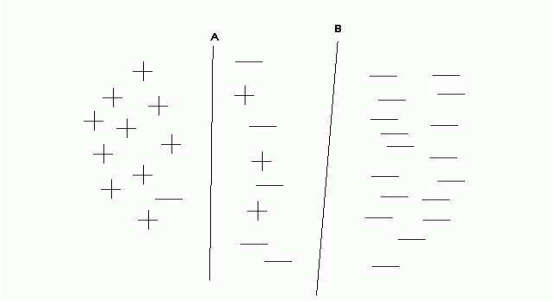


Figure 1: Hyperplane B was trained with a higher $\frac{C_+}{C_-}$ ratio than hyperplane A was trained with.

(14.94% of the examples in AImed are positive). We use the K_{GC} kernel from (Giuliano et al., 2006), one of the highest-performing systems on AImed to date and perform 10-fold cross validation. For TC, we use the Reuters-21578 ModApte split. Since a document may belong to more than one category, each category is treated as a separate binary classification problem, as in (Joachims, 1998). As in (Joachims, 1998), we use the ten largest categories, which have imbalances ranging from 1.88% to 29.96%.

4 AL-SVM Methods for Addressing Class Imbalance

The key question when using cwSVMs is how to set the ratio $\frac{C_+}{C_-}$. Increasing it will typically shift the learned hyperplane so recall is increased and precision is decreased (see Figure 1 for a hypothetical example). Let PA$= \frac{C_+}{C_-}$.[3] How should the PA be set during AL-SVM?

We propose two approaches: one sets the PA based on the level of imbalance in the labeled training data and one aims to set the PA based on an estimate of overall corpus imbalance, *which can drastically differ from the level of imbalance in actively sampled training data*. The first method is called CurrentPA, depicted in Figure 2. Note that in step 0 of the loop, PA is set based on the distribution of positive and negative examples in the *current* set of labeled data. However, observe that during AL the ratio $\frac{\text{\# neg labeled examples}}{\text{\# pos labeled examples}}$ in the current set of labeled data gets skewed from the ratio in the entire

[3]PA stands for positive amplification and gives us a concise way to denote the fraction $\frac{C_+}{C_-}$, which doesn't have a standard name.

Input:

L = small initial set of labeled data

U = large pool of unlabeled data

Loop until stopping criterion is met:

0. Set PA = $\frac{|\{x \in Labeled : f(x) = -1\}|}{|\{x \in L : f(x) = +1\}|}$

where f is the function we desire to learn.

1. Train an SVM with C_+ and C_- set such that $\frac{C_+}{C_-}$ = PA and obtain hyperplane h .[4]

2. $batch \leftarrow$ select k points from U that are closest to h and request their labels.[5]

3. $U = U - batch$.

4. $L = L \cup batch$.

End Loop

Figure 2: The CurrentPA algorithm

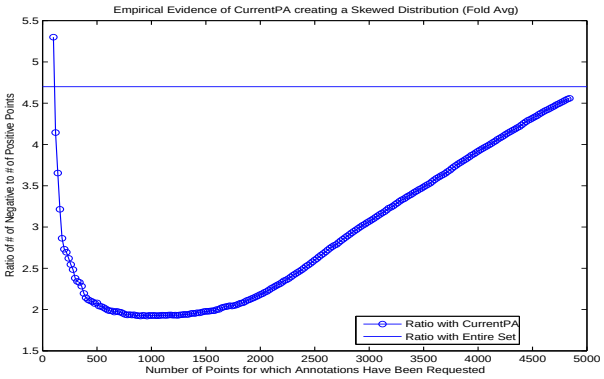


Figure 3: Illustration of AL skewing the distribution of pos/neg points on AImed.

corpus because AL systematically selects the examples that are closest to the current model's hyperplane and this tends to select more positive examples than random selection would select (see also (Ertekin et al., 2007)).

Empirical evidence of this distribution skew is illustrated in Figure 3. The trend toward balanced datasets during AL could mislead and cause us to underestimate the PA.

Therefore, our next algorithm aims to set the PA based on the ratio of neg to pos instances in the entire corpus. However, since we don't have labels for the entire corpus, we don't know this ratio. But by using a small initial sample of labeled data, we can

[4]We use SVM^{light}'s default value for C_-.

[5]In our experiments, batch size is 20.

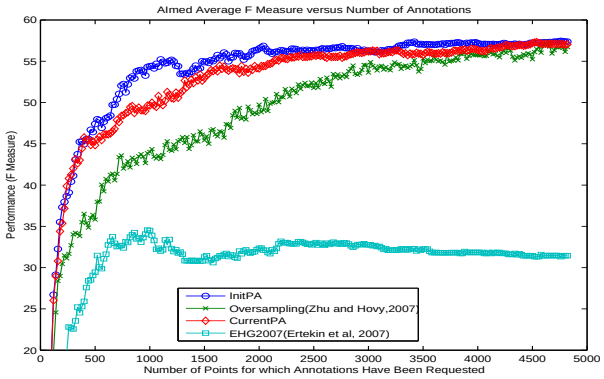


Figure 4: AImed learning curves. y-axis is from 20% to 60%.

estimate this ratio with high confidence. This estimate can then be used for setting the PA throughout the AL process. We call this method of setting the PA based on a small initial set of labeled data the InitPA method. It is like CurrentPA except we move *Step 0* to be executed one time before the loop and then use that same PA value on each iteration of the AL loop.

To guide what size to make the initial set of labeled data, one can determine the sample size required to estimate the proportion of positives in a finite population to within sampling error e with a desired level of confidence using standard statistical techniques found in many college-level statistics references such as (Berenson et al., 1988). For example, carrying out the computations on the AImed dataset shows that a size of 100 enables us to be 95% confident that our proportion estimate is within 0.0739 of the true proportion. In our experiments, we used an initial labeled set of size 100.

5 Evaluation

In addition to InitPA and CurrentPA, we also implemented the methods from (Ertekin et al., 2007; Zhu and Hovy, 2007). We implemented oversampling by duplicating points and by BootOS (Zhu and Hovy, 2007). To avoid cluttering the graphs, we only show the highest-performing oversampling variant, which was by duplicating points. Learning curves are presented in Figures 4 and 5.

Note InitPA is the highest-performing method for all datasets, especially in the practically important area of where the learning curves begin to plateau.

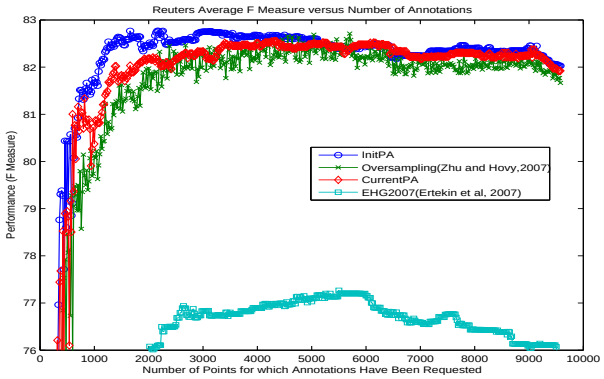


Figure 5: Reuters learning curves. y-axis is from 76% to 83%.

This area is important because this is around where we would want to stop AL (Bloodgood and Vijay-Shanker, 2009).

Observe that the gains of InitPA over CurrentPA are smaller for Reuters. For some Reuters categories, InitPA and CurrentPA have nearly identical performance. Applying the models learned by CurrentPA at each round of AL on the data used to train the model reveals that the recall on the training data is nearly 100% for those categories where InitPA/CurrentPA perform similarly. Increasing the relative penalty for slack error on positive training points will not have much impact if (nearly) all of the pos train points are already classified correctly. Thus, in situations where models are already achieving nearly 100% recall on their train data, InitPA is not expected to outperform CurrentPA.

The hyperplanes learned during AL-SVM serve two purposes: *sampling* - they govern which unlabeled points will be selected for human annotation, and *predicting* - when AL stops, the most recently learned hyperplane is used for classifying test data. Although all AL-SVM approaches we're aware of use the same hyperplane at each round of AL for both of these purposes, this is not required. We compared InitPA with hybrid approaches where hyperplanes trained using an InitPA cost model are used for sampling and hyperplanes trained using a CurrentPA cost model are used for predicting, and vice-versa, and found that InitPA performed better than both of these hybrid approaches. This indicates that the InitPA cost model yields hyperplanes that are better for both sampling and predicting.

6 Conclusions

We've made the case for the importance of AL-SVM for imbalanced datasets and showed that the AL scenario calls for modifications to PL approaches to addressing imbalance. For AL-SVM, the key idea behind InitPA is to base cost models on an estimate of overall corpus imbalance rather than the class imbalance in the so far labeled data. The practical utility of the InitPA method was demonstrated empirically; situations where InitPA won't help that much were made clear; and analysis showed that the sources of InitPA's gains were from both better sampling and better predictive models.

InitPA is an instantiation of a more general idea of *not* using the same inference algorithms during AL as during PL but instead modifying inference algorithms to suit esoteric characteristics of actively sampled data. This is an idea that has seen relatively little exploration and is ripe for further investigation.

References

Mark L. Berenson, David M. Levine, and David Rindskopf. 1988. *Applied Statistics*. Prentice-Hall, Englewood Cliffs, NJ.

Michael Bloodgood and K. Vijay-Shanker. 2009. A method for stopping active learning based on stabilizing predictions and the need for user-adjustable stopping. In *CoNLL*.

Seyda Ertekin, Jian Huang, Léon Bottou, and C. Lee Giles. 2007. Learning on the border: active learning in imbalanced data classification. In *CIKM*.

Claudio Giuliano, Alberto Lavelli, and Lorenza Romano. 2006. Exploiting shallow linguistic information for relation extraction from biomedical literature. In *EACL*.

Thorsten Joachims. 1998. Text categorization with suport vector machines: Learning with many relevant features. In *ECML*, pages 137–142.

Thorsten Joachims. 1999. Making large-scale SVM learning practical. In *Advances in Kernel Methods – Support Vector Learning*, pages 169–184.

Katharina Morik, Peter Brockhausen, and Thorsten Joachims. 1999. Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In *ICML*, pages 268–277.

Jingbo Zhu and Eduard Hovy. 2007. Active learning for word sense disambiguation with methods for addressing the class imbalance problem. In *EMNLP-CoNLL*.

Faster MT Decoding through Pervasive Laziness

Michael Pust and Kevin Knight
Information Sciences Institute
University of Southern California
lastname@isi.edu

Abstract

Syntax-based MT systems have proven effective—the models are compelling and show good room for improvement. However, decoding involves a slow search. We present a new lazy-search method that obtains significant speedups over a strong baseline, with no loss in Bleu.

1 Introduction

Syntax-based string-to-tree MT systems have proven effective—the models are compelling and show good room for improvement. However, slow decoding hinders research, as most experiments involve heavy parameter tuning, which involves heavy decoding. In this paper, we present a new method to improve decoding performance, obtaining a significant speedup over a strong baseline with no loss in Bleu. In scenarios where fast decoding is more important than optimal Bleu, we obtain better Bleu for the same time investment. Our baseline is a full-scale syntax-based MT system with 245m tree-transducer rules of the kind described in (Galley et al., 2004), 192 English non-terminal symbols, an integrated 5-gram language model (LM), and a decoder that uses state-of-the-art cube pruning (Chiang, 2007). A sample translation rule is:

S(x0:NP x1:VP) ↔ x1:VP x0:NP

In CKY string-to-tree decoding, we attack spans of the input string from shortest to longest. We populate each span with a set of edges. An edge contains a English non-terminal (NT) symbol (NP, VP, etc), border words for LM combination, pointers to child edges, and a score. The score is a sum of (1) the left-child edge score, (2) the right-child edge score, (3) the score of the translation rule that combined them, and (4) the target-string LM score. In this paper, we are only concerned with what happens when constructing edges for a single span [i,j]. The naive algorithm works like this:

```
for each split point k
  for each edge A in span [i,k]
    for each edge B in span [k,j]
      for each rule R with RHS = A B
        create new edge for span [i,j]
delete all but 1000-best edges
```

The last step provides a necessary beam. Without it, edges proliferate beyond available memory and time. But even with the beam, the naive algorithm fails, because enumerating all <A,B,R> triples at each span is too time consuming.

2 Cube Pruning

Cube pruning (Chiang, 2007) solves this problem by lazily enumerating triples. To work, cube pruning requires that certain orderings be continually maintained at all spans. First, rules are grouped by RHS into *rule sets* (eg, all the NP-VP rules are in a set), and the members of a given set are sorted by rule score. Second, edges in a span are grouped by NT into *edge sets* (eg, all the NP edges are in an edge set), ordered by edge score.

Consider the sub-problem of building new [i,j] edges by combining (just) the NP edges over [i,k] with (just) the VP edges over [k,j], using the available NP-VP rules. Rather than enumerate all triples, cube pruning sets up a 3-dimensional cube structure whose individually-sorted axes are the NP left edges, the VP right edges, and the NP-VP rules. Because the corner of the cube (best NP left-edge, best VP right-edge, best NP-VP rule) is likely the best edge in the cube, at beam size 1, we would simply return this edge and terminate, without checking other triples. We say "likely" because the corner position does not take into account the LM portion of the score.[1]

After we take the corner and post a new edge from it, we identify its 3 neighbors in the cube. We com-

[1] We also employ LM rule and edge forward-heuristics as in (Chiang, 2007), which improve the sorting.

Proceedings of NAACL HLT 2009: Short Papers, pages 141–144,
Boulder, Colorado, June 2009.

pute their full scores (including LM portion) and push them onto a priority queue (PQ). We then pop an item from the PQ, post another new edge, and push the item's neighbors onto the PQ. Note that this PQ grows in size over time. In this way, we explore the best portion of the cube without enumerating all its contents. Here is the algorithm:

```
push(corner, make-edge(corner)) onto PQ
for i = 1 to 1000
  pop(position, edge) from top of PQ
  post edge to chart
  for each n in neighbors(position)
    push(n, make-edge(n)) onto PQ
  if PQ is empty, break from for-loop
```

The function *make-edge* completely scores an edge (including LM score) before inserting it into the PQ. Note that in practice, we execute the loop up to 10k times, to get 1000 edges that are distinct in their NTs and border words.

In reality, we have to construct many cubes, one for each combinable left and right edge set for a given split point, plus all the cubes for all the other split points. So we maintain a PQ-of-PQs whose elements are cubes.

```
create each cube, pushing its fully-scored corner
  onto the cube's PQ
push cubes themselves onto a PQ-of-PQs
for i = 1 to 1000:
  pop a cube C from the PQ-of-PQs
  pop an item from C
    post edge to chart
    retrieve neighbors, score & push them onto C
  push C back onto the PQ-of-PQs
```

3 Lazy Lists

When we meter the cube pruning algorithm, we find that over 80% of the time goes to building the initial queue of cubes, including deriving a corner edge for each cube—only a small fraction is spent deriving additional edges via exploring the cubes. For spans of length 10 or greater, we find that we have to create more than 1000 cubes, i.e., more than the number of edges we wish to explore.

Our idea, then, is to create the cubes themselves lazily. To describe our algorithm, we exploit an abstract data structure called a *lazy list* (aka generator, stream, pipe, or iterator), which supports three operations:

```
next(list): pops the front item from a list
peek(list): returns the score of the front item
empty(list): returns true if the list is empty
```

A cube is a lazy list (of edges). For our purposes, a lazy list can be implemented with a PQ or something else—we no longer care how the list is populated or maintained, or even whether there are a finite number of elements.

Instead of explicitly enumerating all cubes for a span, we aim to produce a lazy list of cubes. Assume for the moment that such a lazy list exists—we show how to create it in the next section—and call it L. Let us also say that cubes come off L in order of their top edges' scores. To get our first edge, we let C = next(L), and then we call next(C). Now a question arises: do we pop the next-best edge off C, or do we investigate the next cube in L? We can decide by calling peek(peek(L)). If we choose to pop the next cube (and then its top edge), then we face another (this time three-way) decision. Bookkeeping is therefore required if we are to continue to emit edges in a good order.

We manage the complexity through the abstraction of a *lazy list of lazy lists*, to which we routinely apply a single, key operation called *merge-lists*. This operation converts a lazy list of lazy lists of X's into a simple lazy list of X's. X can be anything: edges, integers, lists, lazy lists, etc.

Figure 1 gives the generic merge-lists algorithm. The *yield* function suspends computation and returns to the caller. peek() lets the caller see what is yielded, next() returns what is yielded and resumes the loop, and empty() tells if the loop is still active.

We are now free to construct any nested "list of lists of lists ... of lists of X" (all lazy) and reduce it stepwise and automatically to a single lazy list. Standard cube pruning (Section 2) provides a simple example: if L is a list of cubes, and each cube is a lazy list of edges, then merge-lists(L) returns us a lazy list of edges (M), which is exactly what the decoder wants. The decoder can populate a new span by simply making 1000 calls to next(M).

4 Pervasive Laziness

Now we describe how to generate cubes lazily. As with standard cube pruning, we need to maintain a

merge-lists(L):
(L is a lazy list of lazy lists)
1. set up an empty PQ of lists, prioritized by peek(list)
2. push next(L) onto PQ
3. pop list L2 off PQ
4. yield pop(L2)
5. if !empty(L2) and peek(L2) is worse than peek(peek(L)), then push next(L) onto PQ
6. if !empty(L2), then push L2 onto PQ
7. go to step 3

Figure 1: Generic merge-lists algorithm.

small amount of ordering information among edges in a span, which we exploit in constructing higher-level spans. Previously, we required that all NP edges be ordered by score, the same for VP edges, etc. Now we additionally order whole *edge sets* (groups of edges sharing an NT) with respect to each other, eg, NP $>$ VP $>$ RB $>$ etc. These are ordered by the top-scoring edges in each set.

Ideally, we would pop cubes off our lazy list in order of their top edges. Recall that the PQ-of-PQs in standard cube pruning works this way. We cannot guarantee this anymore, so we approximate it.

Consider first a single edge set from [i,k], eg, all the NP edges. We build a lazy list of cubes that all have a left-NP. Because edge sets from [k,j] are ordered with respect to each other, we may find that it is the VP edge set that contains the best edge in [k,j]. Pulling in all NP-VP rules, we can now postulate a "best cube," which generates edges out of left-NPs and right-VPs. We can either continue making edge from this cube, or we can ask for a "second-best cube" by moving to the next edge set of [k,j], which might contain all the right-PP edges. Thus, we have a lazy list of left-NP cubes. Its ordering is approximate—cubes come off in such a way that their top edges go from best to worst, but only considering the left and right child scores, not the rule scores. This is the same idea followed by standard cube pruning when it ignores internal LM scores.

We next create similar lazy lists for all the other [i,k] edge sets (not just NP). We combine these lists into a higher-level lazy list, whose elements pop off according to the ordering of edge sets in [i,k]. This structure contains all edges that can be produced

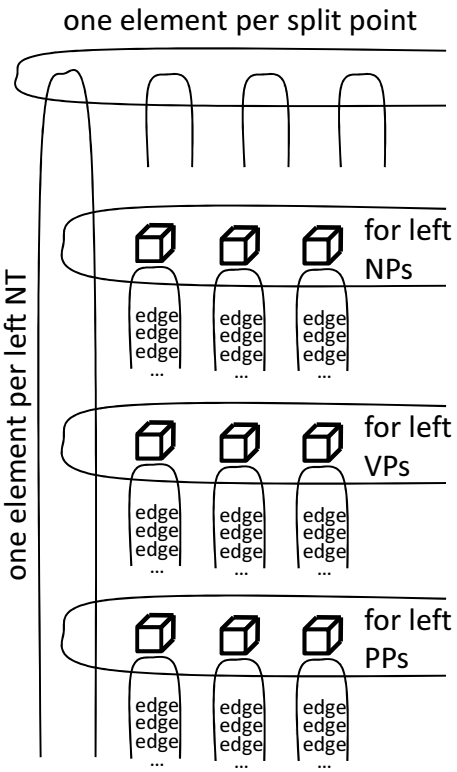


Figure 2: Organizing lazy lists for the decoder.

from split point k. We call merge-lists recursively on the structure, leaving us with a single lazy list M of edges. The decoder can now make 1000 calls to next(M) to populate the new span.

Edges from other split points, however, must compete on an equal basis for those 1000 slots. We therefore produce a separate lazy list for each of the $j - i - 1$ split points and combine these into an even higher-level list. Lacking an ordering criterion among split points, we presently make the top list a non-lazy one via the PQ-of-PQs structure. Figure 2 shows how our lists are organized.

The quality of our 1000-best edges can be improved. When we organize the higher-level lists by left edge-sets, we give prominence to the best left edge-set (eg, NP) over others (eg, VP). If the left span is relatively short, the contribution of the left NP to the total score of the new edge is small, so this prominence is misplaced. Therefore, we repeat the above process with the higher-level lists organized by right span instead of left. We merge the right-oriented and left-oriented structures, making sure that duplicates are avoided.

Related Work. Huang and Chiang (2007) de-

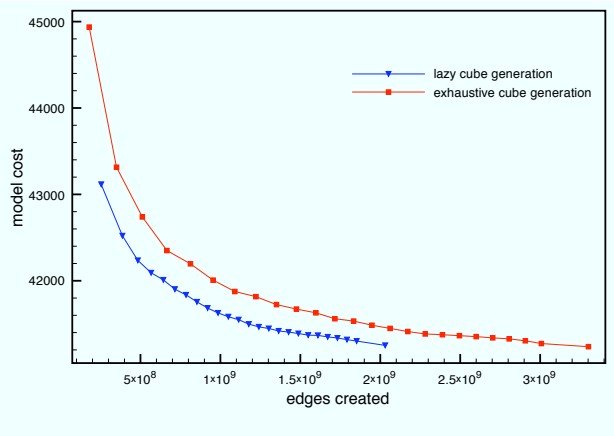


Figure 3: Number of edges produced by the decoder, versus model cost of 1-best decodings.

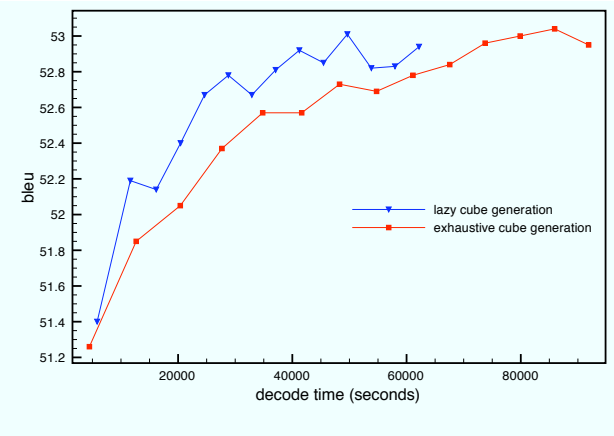


Figure 4: Decoding time versus Bleu.

scribe a variation of cube pruning called cube growing, and they apply it to a source-tree to target-string translator. It is a two pass approach, where a context-free parser is used to build a source forest, and a top down lazy forest expansion is used to integrate a language model. The expansion recursively calls cubes top-down, in depth first order. The context-free forest controls which cubes are built, and acts as a heuristic to minimize the number of items returned from each cube necessary to generate k-best derivations at the top.

It is not clear that a decoder such as ours, without the source-tree constraint, would benefit from this method, as building a context-free forest consistent with future language model integration via cubes is expensive on its own. However, we see potential integration of both methods in two places: First, the merge-lists algorithm can be used to lazily process any nested for-loops—including vanilla CKY—provided the iterands of the loops can be prioritized. This could speed up the creation of a first-pass context-free forest. Second, the cubes themselves could be prioritized in a manner similar to what we describe, using the context-free forest to prioritize cube generation rather than antecedent edges in the chart (since those do not exist yet).

5 Results

We compare our method with standard cube pruning (Chiang, 2007) on a full-scale Arabic/English syntax-based MT system with an integrated 5-gram LM. We report on 500 test sentences of lengths 15-35. There are three variables of interest: runtime, model cost (summed across all sentences), and IBM Bleu. By varying the beam sizes (up to 1350), we obtain curves that plot edges-produced versus model-cost, shown in Figure 3. Figure 4 plots Bleu score against time. We see that we have improved the way our decoder searches, by teaching it to explore fewer edges, without sacrificing its ability to find low-cost edges. This leads to faster decoding without loss in translation accuracy.

Taken together with cube pruning (Chiang, 2007), k-best tree extraction (Huang and Chiang, 2005), and cube growing (Huang and Chiang, 2007), these results provide evidence that lazy techniques may penetrate deeper yet into MT decoding and other NLP search problems.

We would like to thank J. Graehl and D. Chiang for thoughts and discussions. This work was partially supported under DARPA GALE, Contract No. HR0011-06-C-0022.

References

D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2).

M. Galley, M. Hopkins, K. Knight, and D. Marcu. 2004. What's in a translation rule. In *Proc. NAACL-HLT*.

L. Huang and D. Chiang. 2005. Better k-best parsing. In *Proc. IWPT*.

L. Huang and D. Chiang. 2007. Forest rescoring: Faster decoding with integrated language models. In *Proc. ACL*.

Evaluating the Syntactic Transformations in Gold Standard Corpora for Statistical Sentence Compression

Naman K. Gupta, Sourish Chaudhuri, Carolyn P. Rosé
Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{nkgupta,sourishc,cprose}@cs.cmu.edu

Abstract

We present a policy-based error analysis approach that demonstrates a limitation to the current commonly adopted paradigm for sentence compression. We demonstrate that these limitations arise from the strong assumption of locality of the decision making process in the search for an acceptable derivation in this paradigm.

1 Introduction

In this paper we present a policy-based error analysis approach that demonstrates a limitation to the current commonly adopted paradigm for sentence compression (Knight and Marcu, 2000; Turner and Charniak, 2005; McDonald, 2006; Clark and Lapata 2006).

Specifically, in typical statistical compression approaches, a simplifying assumption is made that compression is accomplished strictly by means of word deletion. Furthermore, each sequence of contiguous words that are dropped from a source sentence is considered independently of other sequences of words dropped from other portions of the sentence, so that the features that predict whether deleting a sequence of words is preferred or not is based solely on local considerations. This simplistic approach allows all possible derivations to be modeled and decoded efficiently within the search space, using a dynamic programming algorithm.

In theory, it should be possible to learn how to generate effective compressions using a corpus of source-target sentence pairs, given enough examples and sufficiently expressive features. However, our analysis casts doubt that this framework with its strong assumptions of locality is sufficiently powerful to learn the types of example compressions frequently found in corpora of human generated gold standard compressions regardless of how expressive the features are.

Work in sentence compression has been somewhat hampered by the tremendous cost involved in producing a gold standard corpus. Because of this tremendous cost, the same gold standard corpora are used in many different published studies almost as a black box. This is done with little scrutiny of the limitations on the learnability of the desired target systems. These limitations result from inconsistencies due to the subtleties in the process by which humans generate the gold standard compressions from the source sentences, and from the strong locality assumptions inherent in the frameworks.

Typically, the humans who have participated in the construction of these corpora have been instructed to preserve grammaticality and to produce compressions by deletion. Human ratings of the gold standard compressions by separate judges confirm that the human developers have literally followed the instructions, and have produced compressions that are themselves largely grammatical. Nevertheless, what we demonstrate with our error analysis is that they have used meaning preserving transformation that didn't consistently preserve the grammatical relations from the source sentence while transforming source sentences into target sentences. This places limitations on how well the preferred patterns of compression can be learned using the current paradigm and existing corpora.

In the remainder of the paper, we discuss relevant work in sentence compression. We then introduce our policy-based error analysis technique. Next we discuss the error analysis itself and the conclusions we draw from it. Finally, we conclude

Proceedings of NAACL HLT 2009: Short Papers, pages 145–148,
Boulder, Colorado, June 2009.

with future directions for broader application of this error analysis technique.

2 Related Work

Knight and Marcu (2000) present two approaches to the sentence compression problem- one using a noisy channel model and the other using a decision-based model. Subsequent work (McDonald, 2006) has demonstrated an advantage for a soft constraint approach, where a discriminative model learns to make local decisions about dropping a sequence of words from the source sentence in order to produce the target compression. Features in this system are defined over pairs of words in the source sentence, with the idea that the pair of words would appear adjacent in the resulting compression, with all intervening words dropped. Thus, the features represent this transformation, and the feature weights are meant to indicate whether the transformation is associated with good compressions or not.

We use McDonald's (2006) proposed model as a foundation for our work because its soft constraint approach allows for natural integration of a variety of classes of features, even overlapping features. In our prior work we have explored the potential for improving the performance of a compression system by including additional, more sophisticated syntactically motivated features than those included in previously published models. In this paper, we evaluate the gold standard corpus itself using similar syntactic grammar policies.

3 Grammar Policy Extraction

In the domain of Sentence Compression, the corpus consists of source sentences each paired with a gold standard compressed sentence. Most of the above related work has been evaluated using the following 2 corpora, namely the Ziff-Davis (ZD) set (Knight and Marcu, 2002) consisting of 1055 sentences, and a partial Broadcast News Corpus (CL Corpus) (Clarke and Lapata, 2006) originally consisting of 1619 sentences, of which we used 1070 as the training set in our development work as well as in the error analysis below. Hence, we use these two popular corpora to present our work. We hypothesize certain grammar policies that intuitively should be followed while deriving the target-compressed sentence from the source sentence if the mapping between source and target sentences is produced via grammatical transformations. The basic idea behind these policies grows out of the same ideas motivating the syntactic features used in McDonald (2006). These policies, extracted using the MST (McDonald, 2005) dependency parse structure of the source sentence, are as follows:

1. The syntactic root word of a sentence should be retained in the compressed sentence.
2. If a verb is retained in the compressed sentence, then the dependent subject of that verb should also be retained.
3. If a verb is retained in the compressed sentence, then the dependent object of that verb should also be retained.
4. If the verb is dropped in the compressed sentence then its arguments, namely subject, object, prepositional phrases etc., should also be dropped.
5. If the Preposition in a Prepositional phrase (PP) is retained in the compressed sentence, then the dependent Noun Phrase (NP) of that Preposition should also be retained.
6. If the head noun of a Noun phrase (NP) within a Prepositional phrase is retained in the compressed sentence, then the syntactic parent Preposition of the NP should also be retained.
7. If a Preposition, the syntactic head of a Prepositional phrase (PP), is dropped in the compressed sentence, then the whole PP, including dependent Noun phrase in that PP, should also be dropped.
8. If the head noun of a Noun phrase within a Prepositional phrase (PP) is dropped in the compressed sentence, then the syntactic parent Preposition of the PP should also be dropped.

These grammar policies make predictions about where, in the phrase structure, constituents are likely to be dropped or retained in the compression. Thus, these policies have similar motivation to the syntactic features in the McDonald (2006) model. However, there is a fundamental difference in the way these policies are computed. In the McDonald (2006) model, the features are com-

puted locally over adjacent words y_{i-1} & y_i in the compression and the words dropped from the original sentence between that word range y_{i-1} & y_i. In cases where the syntactic structure of the involved words extends beyond this range, the extracted features are not able to capture all of the relevant syntactic dependencies. On the other hand, in our analysis the policies are computed globally over the complete sentence without specifying any range of words. As an illustrative example, let us consider the following sentence from the CL Corpus (bold represents dropped words):

1. The$_1$ leaflet$_2$ given$_3$ to$_4$ Labour$_5$ **activists$_6$** mentions$_7$ none$_8$ of$_9$ these$_{10}$ things$_{11}$**.**

According to Policy 2, since the verb 'mentions' is retained, the subject of the verb 'the leaflet' should also be retained. In the McDonald (2006) model, by looking at the local range $y_{i-1} = 5$ and $y_i = 7$ for the verb 'mentions', we will not be able to compute whether the subject(1,2) was retained in the compression or not. So this policy can be captured only if the global context is taken into account while evaluating the verb 'mentions'.

Now we evaluate each sentence in the corpus to determine whether a particular policy was applicable and if applicable then whether it was violated. Table 1 shows the summary of the evaluation of all the sentences in the two corpora. Column 2 in the table shows the percentage of sentences in the ZD Corpus where the respective policies were applicable. And column 3 shows the percentage of sentences where the respective policies were violated, whenever applicable. Columns 4 and 5 show respective percentages for the CL corpus.

4 Evaluation

In this section we discuss the results from evaluating the 8 grammar policies discussed in Section 3 over the ZD and CL corpora, as discussed above.

The policies were evaluated with respect to whether they applied in a sentence, i.e., whether the premise of the "if … then" rule is true in the sentence, and whether the policy was broken when applied, i.e., if the premise is true but the consequent is false. The striking finding is that for every one of the policies discussed in the previous section, they are violated for at least 10% of the sentences where they applied, and sometimes as much as 72%. For most policies, the proportion of sentences where the policy is violated when applied is a minority of cases. Thus, based on this, we can expect that grammar oriented features motivated by these policies and derived from a syntactic analysis of the source and/or target sentences in the gold standard could be used to improve the performance of compression systems that don't make use of syntactic information to that extent. However, the noticeable proportion of violations with respect to some of the policies indicate that there is a limited extent to which these types of features can contribute towards improved performance.

One observation we make from Table 1 is that while the proportion of sentences where the policies (Columns 2 and 4) apply as well as the proportion of sentences where the policies are broken when applied (Columns 3 and 5) are highly correlated between the two corpora. Nevertheless, the distributions are not identical. Thus, again, while we predict that using this style of dependency syntax features might improve performance of compression systems within a single corpus, we would not expect trained models that rely on these syntactic dependency features to generalize in an ideal way between corpora.

	ZD (% Applicable)	ZD (% Violations when Applicable)	CL (% Applicable)	CL (% Violations when Applicable)
Policy1	100%	34%	100%	14%
Policy2	66%	18%	84%	18%
Policy3	50%	10%	61%	24%
Policy4	59%	59%	46%	72%
Policy5	62%	17%	77%	27%
Policy6	65%	22%	79%	29%
Policy7	57%	25%	58%	40%
Policy8	55%	16%	58%	36%

Table 1: Summary of evaluation of grammar policies over the Ziff-Davis (ZD) training set and Clark-Lapata (CL) training set.

Beyond the above evaluation illustrating the extent to which grammar inspired policies are violated in human generated gold standard corpora, interesting insights into challenges that must be addressed in order to improve performance can be obtained by taking a close look at typical examples from the CL corpus where the policies are broken in the

gold standard corpora (bold represents dropped words).

1. The attempt to **put flesh and blood on the skeleton** structure **of** a **possible** united Europe emerged.
2. Annely **has used the gallery** 's three floors **to** divide the exhibits into three **distinct** groups.
3. Labor **has said it** will scrap the system.
4. Montenegro 's **sudden** rehabilitation of Nicholas **'s memory** is **a** popular **move**.

In Sentence 1, retaining the dependent Noun *structure* of the dropped Preposition *on* in the PP violates Policy 7. Such a NP to Infinitive Phrase transformation changes the syntactic structure of the sentence. Sentence 2 also breaks several policies, namely Policies 1, 4 and 7. The syntactic root *has* is dropped. Also the main verb *has used* is dropped while retaining the Subject *Annely*. In Sentence 3, breaking Policies 1, 2 and 4, the human annotators replaced the pronoun *it* with the noun *Labor*, the subject of a dropped verb 'has said'. Such anaphora resolution cannot be done without relevant context, which is not available in strictly local paradigms of sentence compression. In Sentence 4, policies 3. 5 and 8 are violated. Transformations like substituting *Nicholas's memory* by the metonym *Nicholas* and *popular move* by *popular* need to be identified and analyzed. Such varied transformations, made in the syntactic structure of the sentences by human annotators, are counter-intuitive, making them hard to be captured in the linear models learned in association with the syntactic features in current compression systems.

5 Conclusions and Current Directions

In this paper we have introduced a policy-based error analysis technique that was used to investigate the potential impact and limitations of adding a particular style of dependency parse features to typical statistical compression systems. We have argued that the reason for the limitation arises from the strong assumption of the local nature of the decisions that are made in obtaining the system-generated compression from a source sentence.

Other related technologies such as statistical machine translation and statistical paraphrase are based on similar paradigms with similar assumptions of the local nature of decisions that are made in the search for an acceptable derivation. We conjecture both that it is likely that the same issues related to the construction of the gold standard corpora likely apply and that a similar policy-based error analysis approach could be used in order to assess the extent to which this is true and identify possible directions for improving performance. In our ongoing work, we plan to conduct a similar error analysis for these problems in order to evaluate the generality of the findings reported here.

Acknowledgments

This work was funded in part by the Office of Naval Research grant number N00014510043.

References

James Clarke and Mirella Lapata. 2006. *Constraint-Based Sentence Compression: An Integer Programming Approach.* Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions (ACL-2006), pages 144-151, 2006.

James Clarke and Mirella Lapata. 2006. *Models for Sentence Compression: A Comparison across Domains, Training Requirements and Evaluation Measures.* Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics, 377-384. Sydney, Australia.

Kevin Knight and Daniel Marcu. 2000. *Statistics-Based Summarization – Step One: Sentence Compression.* Proceedings of AAAI-2000, Austin, TX, USA.

Knight, Kevin and Daniel Marcu. 2002. *Summarization beyond sentence extraction: a probabilistic approach to sentence compression*. Artificial Intelligence 139(1):91–107.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. *Online large-margin training of dependency parsers.* Proc. ACL.

Ryan Mcdonald, 2006. *Discriminative sentence compression with soft syntactic constraints*. Proceedings of the 11th EACL. Trento, Italy, pages 297--304.

Jenine Turner and Eugene Charniak. 2005. *Supervised and unsupervised learning for sentence compression*. Proc. ACL.

Incremental Adaptation of Speech-to-Speech Translation

Nguyen Bach, Roger Hsiao, Matthias Eck, Paisarn Charoenpornsawat, Stephan Vogel, Tanja Schultz, Ian Lane, Alex Waibel and Alan W. Black
InterACT, Language Technologies Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{nbach, wrhsiao, matteck, paisarn, stephan.vogel, tanja, ianlane, ahw, awb}@cs.cmu.edu

Abstract

In building practical two-way speech-to-speech translation systems the end user will always wish to use the system in an environment different from the original training data. As with all speech systems, it is important to allow the system to adapt to the actual usage situations. This paper investigates how a speech-to-speech translation system can adapt day-to-day from collected data on day one to improve performance on day two. The platform is the CMU Iraqi-English portable two-way speech-to-speech system as developed under the DARPA TransTac program. We show how machine translation, speech recognition and overall system performance can be improved on day 2 after adapting from day 1 in both a supervised and unsupervised way.

1 Introduction

As speech-to-speech translation systems move from the laboratory into field deployment, we quickly see that mismatch in training data with field use can degrade the performance of the system. Retraining based on field usage is a common technique used in all speech systems to improve performance. In the case of speech-to-speech translation we would particularly like to be able to adapt the system based on its usage automatically without having to ship data back to the laboratory for retraining. This paper investigates the scenario of a two-day event. We wish to improve the system for the second day based on the data collected on the first day.

Our system is designed for eyes-free use and hence provides no graphical user interface. This allows the user to concentrate on his surrounding environment during an operation. The system only provides audio control and feedback. Additionally the system operates on a push-to-talk method. Previously the system (Hsiao et al., 2006; Bach et al., 2007) needed 2 buttons to operate, one for the English speaker and the other one for the Iraqi speaker.

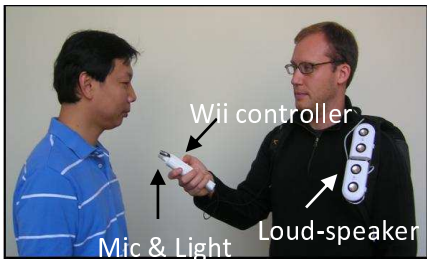


Figure 1: The users interact with the system

To make the system easier and faster to use, we propose to use a single button which can be controlled by the English speaker. We mounted a microphone and a Wii remote controller together as shown in 1.

Since the Wii controller has an accelerometer which can be used to detect the orientation of the controller, this feature can be applied to identify who is speaking. When the English speaker points towards himself, the system will switch to English-Iraqi translation. However, when the Wii is pointed towards somebody else, the system will switch to Iraqi-English translation. In addition, we attach a light on the Wii controller providing visual feedback. This can inform an Iraqi speaker when to start speaking. The overall system is composed of five major components: two automatic speech recognition (ASR) systems, a bidirectional statistical machine translation (SMT) system and two text-to-speech (TTS) systems.

2 Data Scenario

The standard data that is available for the TransTac project was collected by recording human interpreter mediated dialogs between war fighters and Iraqi native speakers in various scenarios. The dialog partners were aware that the data was being collected for training machine based translation devices, but would often talk directly to the human interpreter rather than pretending it was an automatic device. This means that the dialog

Proceedings of NAACL HLT 2009: Short Papers, pages 149–152,
Boulder, Colorado, June 2009.

partners soon ignored the recording equipment and used a mostly natural language, using informal pronunciation and longer sentences with more disfluencies than we find in machine mediated translation dialogs.

Most users mismatch their language when they communicate using an automatic speech-to-speech translation system. They often switch to a clearer pronunciation and use shorter and simpler sentences with less disfluency. This change could have a significant impact on speech recognition and machine translation performance if a system was originally trained on data from the interpreter mediated dialogs.

For this reason, additional data was collected during the TransTac meeting in June of 2008. This data was collected with dialog partners using the speech-to-speech translation systems from 4 developer participants in the TransTac program. The dialog partners were given a description of the specific scenario in form of a rough script and had to speak their sentences into the translation systems. The dialog partners were not asked to actually react to the potentially incorrect translations but just followed the script, ignoring the output of the translation system. This has the effect that the dialog partners are no longer talking to a human interpreter, but to a machine, pressing push-to-talk buttons etc. and will change their speech patterns accordingly.

The data was collected over two days, with around 2 hours of actual speech per day. This data was transcribed and translated, resulting in 864 and 824 utterance pairs on day 1 and 2, respectively.

3 ASR LM Adaptation

This section describes the Iraqi ASR system and how we perform LM adaptation on the day 1 data to improve ASR performance on day 2. The CMU Iraqi ASR system is trained with around 350 hours of audio data collected under the TransTac program. The acoustic model is speaker independent but incremental unsupervised MLLR adaptation is performed to improve recognition. The acoustic model has 6000 codebooks and each codebook has at most 64 Gaussian mixtures determined by merge-and-split training. Semi-tied covariance and boosted MMI discriminative training is performed to improve the model (Povey et al., 2009). The features for the acoustic model is the standard 39-dimension MFCC and we concatenate adjacent 15 frames and perform LDA to reduce the dimension to 42 for the final feature vectors. The language model of the ASR system is a trigram LM trained on the audio transcripts with around three million words with Kneser-Ney smoothing (Stolcke, 2002).

To perform LM adaptation for the ASR system, we use the ASR hypotheses from day 1 to build a LM. This LM is then interpolated with the original trigram LM to produce an adapted LM for day 2. We also evaluate the effect of having transcribers provide accurate transcription references for day 1 data, and see how it may improve the performance on day 2. We compare unigram, bigram and trigram LMs for adaptation. Since the amount of day 1 data is much smaller than the whole training set and we do not assume transcription of day 1 is always available, the interpolation weight is chosen of be 0.9 for the original trigram LM and 0.1 for the new LM built from the day 1 data. The WER of baseline ASR system on day 1 is 32.0%.

Base	1-g hypo	2-g hypo	3-g hypo	1-g ref	2-g ref	3-g ref
31.3	30.9	31.2	31.1	30.6	30.5	30.4

Table 1: Iraqi ASR's WER on day 2 using different adaptation schemes for day 1 data

The results in Table 1 show that the ASR benefits from LM adaptation. Adapting day 1 data can slightly improve the performance of day 2. The improvement is larger when day 1 transcript is available which is expected. The result also shows that the unigram LM is the most robust model for adaptation as it works reasonably well when transcripts are not available, whereas bigram and trigram LM are more sensitive to the ASR errors made on day 1.

	Day 1	Day 2
No ASR adaptation	29.39	27.41
Unsupervised ASR adaptation	31.55	27.66
Supervised ASR adaptation	32.19	27.65

Table 2: Impact of ASR adaptation to SMT

Table 2 shows the impact of ASR adaptation on the performance of the translation system in BLEU (Papineni et al., 2002). In these experiments we only performed adaptation on ASR and still using the baseline SMT component. There is no obvious difference between unsupervised and supervised ASR adaptation on performance of SMT on day 2. However, we can see that the difference in WER on day 2 of unsupervised and supervised ASR adaptation is relatively small.

4 SMT Adaptation

The Iraqi-English SMT system is trained with around 650K sentence pairs collected under the TransTac program. We used PESA phrase extraction (Vogel, 2005) and a suffix array language model (Zhang and Vogel, 2005). To adapt SMT components one approach is to optimize LM interpolation weights by minimizing perplexity of the 1-best translation output (Bulyko et al., 2007). Related work including (Eck et al., 2004) attempts to use information retrieval to select training sentences similar to those in the test set. To adapt the SMT components we use a domain-specific LM on top of the background

language models. This approach is similar to the work in (Chen et al., 2008). sThe adaptation framework is 1) create a domain-specific LM via an n-best list of day 1 machine translation hypothesis, or day 1 translation references; 2) re-tune the translation system on day 1 via minimum error rate training (MERT) (Venugopal and Vogel, 2005).

Use		Day 1	Day 2
	Baseline	29.39	**27.41**
500 Best MT Hypos	1gramLM	29.18	27.23
	2gramLM	29.53	27.50
	3gramLM	29.36	27.23

Table 3: Performance in BLEU of unsupervised adaptation.

The first question we would like to address is whether our adaptation obtains improvements via an unsupervised manner. We take day 1 baseline ASR hypothesis and use the baseline SMT to get the MT hypothesis and a 500-best list. We train a domain LM using the 500-best list and use the MT hypotheses as the reference in MERT. We treat day 1 as a development set and day 2 as an unseen test set. In Table 3 we compare the performance of four systems: the baseline which does not have any adaptation steps; and 3 adapted systems using unigram, bigram and trigram LMs build from 500-best MT hypotheses.

Use		Day 1	Day 2
	Baseline (no tune)	29.39	27.41
	Baseline (tune)	29.49	27.30
500 Best MT Hypos	1gramLM	30.27	**28.29**
	2gramLM	30.39	**28.30**
	3gramLM	28.36	24.64
MT Ref	1gramLM MT Ref	30.53	**28.35**

Table 4: Performance in BLEU of supervised adaptation.

Experimental results from unsupervised adaptation did not show consistent improvements but suggest we may obtain gains via supervised adaptation. In supervised adaptation, we assume we have day 1 translation references. The references are used in MERT. In Table 4 we show performances of two additional systems which are the baseline system without adaptation but tuned toward day 1, and the adapted system which used day 1 translation references to train a unigram LM (1gramLM MT Ref). The unigram and bigram LMs from 500-best and unigram LM from MT day 1 references perform relatively similar on day 2. Using a trigram 500-best LM returned a large degradation and this LM is sensitive to the translation errors on day1

5 Joint Adaptation

In Sections 3 and 4 we saw that individual adaptation helps ASR to reduce WER and SMT to increase BLEU score. The next step in validating the adaptation framework was to check if the joint adaptation of ASR and SMT on day 1 data will lead to improvements on day 2. Table 5 shows the combination of ASR and SMT adaptation methods. Improvements are obtained by using both ASR and SMT adaptation. Joint adaptation consistently gained more than one BLEU point improvement on day 2. Our best system is unsupervised ASR adaptation via 1gramLM of ASR day 1 transcription coupled with supervised SMT adaptation via 1gramLM of day 1 translation references. An interesting result is that to have a better result on day 2 our approach only requires translation references on day 1. We selected 1gramLM of 500-best MT hypotheses to conduct the experiments since there is no significant difference between 1gramLM and 2gramLM on day 2 as showed in Table 3.

ASR	SMT	Day 1	Day 2
No adaptation	No adaptation	29.39	27.41
Unsupervised ASR adaptation with 1gramLM ASR hypo	1gramLM 500-Best MT Hypo	32.07	28.65
	1gramLM MT Ref	31.76	**28.83**
Supervised ASR adaptation with 1gramLM transcription	1gramLM 500-Best MT Hypo	32.48	28.59
	1gramLM MT Ref	32.68	28.60

Table 5: Performance in BLEU of joint adaptation.

6 Selective Adaptation

The previous results indicate that we require human translation references on day 1 data to get improved performance on day 2. However, our goal is to make a better system on day 2 but try to minimize human efforts on day 1. Therefore, we raise two questions: 1) Can we still obtain improvements by not using all of day 1 data? and 2) Can we obtain more improvements?

To answer these questions we performed oracle experiments when we take the translation hypotheses on day 1 of the baseline SMT and compare them with translation references, then select sentences which have BLEU scores higher than a threshold. The subset of day 1 sentences is used to perform supervised adaptation in a similar way showed in section 5. These experiments also simulate the situation when we have a perfect confidence score for machine translation hypothesis selection. Table 6 shows results when we use various portions of day 1 to perform adaptation. By using day 1 sentences which have smoothed sentence BLEU scores higher than 10 or 20 we have very close performance with adaptation by using all day 1 data. The results also show that by using 416 sentences which have sentence BLEU score higher than 40 on day 1, our adapted translation components outperform the baseline. Performance starts degrading after 50. Experimental results lead to the answer for question 1) that

by using less day 1 data our adapted translation components still obtain improvements compare with the baseline, and 2) we did not see that using less data will lead us to a better performance compare with using all day 1 data.

	No. sents	Day 1	Day 2
Baseline		29.39	27.41
≥ 0	864	30.27	28.29
≥ 10	797	31.15	**28.27**
≥ 20	747	30.81	**28.24**
≥ 30	585	30.04	**27.71**
≥ 40	416	29.72	**27.65**
≥ 50	296	30.06	27.04
Correct	98	29.18	27.19

Table 6: Performance in BLEU of selective adaptation

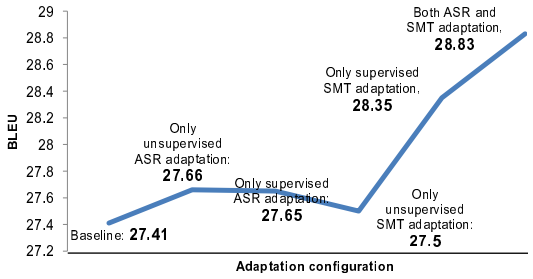


Figure 2: Summarization of adaptation performances

7 Conclusions

This work clearly shows that improvement is possible using collected data for adaptation. The overall picture is shown in Figure 2. However this result is only based on one such data set, it would be useful to do such adaptation over multiple days. The best results however still require producing translation references, notably ASR transcriptions do not seem to help, but may still be required in the process of generating translation references. We wish to further investigate automatic adaptation based on implicit confidence scores, or even active participation of the user e.g. by marking bad utterance which could be excluded from the adaptation.

Acknowledgments

This work is in part supported by the US DARPA under the TransTac (Spoken Language Communication and Translation System for Tactical Use) program. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of DARPA. We would also like to thank Cepstral LLC and Mobile Technologies LLC, for support of some of the lower level software components.

References

Nguyen Bach, Matthias Eck, Paisarn Charoenpornsawat, Thilo Khler, Sebastian Stker, ThuyLinh Nguyen, Roger Hsiao, Alex Waibel, Stephan Vogel, Tanja Schultz, and Alan Black. 2007. The CMU TransTac 2007 Eyes-free and Hands-free Two-way Speech-to-Speech Translation System. In *Proc. of the International Workshop on Spoken Language Translation*, Trento, Italy.

Ivan Bulyko, Spyros Matsoukas, Richard Schwartz, Long Nguyen, and John Makhoul. 2007. Language Model Adaptation in Machine Translation from Speech. In *Proc. of Int. Conf. on Acoustics, Speech and Signal Processing*, Honolulu, Hawaii, USA.

Boxing Chen, Min Zhang, Aiti Aw, and Haizhou Li. 2008. Exploiting n-best hypotheses for smt self-enhancement. In *Proceedings of ACL-08: HLT, Short Papers*, pages 157–160, Columbus, Ohio, USA, June.

Matthias Eck, Stephan Vogel, and Alex Waibel. 2004. Language model adaptation for statistical machine translation based on information retrieval. In *Proc. LREC'04*, Lisbon, Portugal.

Roger Hsiao, Ashish Venugopal, Thilo Kohler, Ying Zhang, Paisarn Charoenpornsawat, Andreas Zollmann, Stephan Vogel, Alan W Black, Tanja Schultz, and Alex Waibel. 2006. Optimizing Components for Handheld Two-way Speech Translation for an English-Iraqi Arabic System. In *Proc. of Interspeech*, Pittsburgh, USA.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL'02*, pages 311–318, Philadelphia, PA, July.

Daniel Povey, Dimitri Kanevsky, Brian Kingsbury, Bhu vana Ramabhadran, George Saon, and Karthik Visweswariah. 2009. Boosted MMI for model and feature-space discriminative training. In *Proc. of Int. Conf. on Acoustics, Speech and Signal Processing*, Las Vegas, USA.

Andreas Stolcke. 2002. SRILM – An extensible language modeling toolkit. In *Proc. Intl. Conf. on Spoken Language Processing*, volume 2, pages 901–904, Denver.

Ashish Venugopal and Stephan Vogel. 2005. Considerations in maximum mutual information and minimum classification error training for statistical machine translation. In *Proceedings of EAMT-05*, Budapest, Hungary.

Stephan Vogel. 2005. Pesa: Phrase pair extraction as sentence splitting. In *Proc. of MT SUMMIT X*, Phuket, Thailand.

Ying Zhang and Stephan Vogel. 2005. An efficient phrase-to-phrase alignment model for arbitrarily long phrase and large corpora. In *Proceedings of EAMT'05*, Budapest, Hungary, May. The European Association for Machine Translation.

Name Perplexity

Octavian Popescu

Abstract

The accuracy of a Cross Document Coreference system depends on the amount of context available, which is a parameter that varies greatly from corpora to corpora. This paper presents a statistical model for computing name perplexity classes. For each perplexity class, the prior probability of coreference is estimated. The amount of context required for coreference is controlled by the prior coreference probability. We show that the prior probability coreference is an important factor for maintaining a good balance between precision and recall for cross document coreference systems.

1 Introduction

The Person Cross Document Coreference (PCDC) task which requires that all and only the textual mentions of an entity of type Person be individuated in a large collection of text documents, is a challenging tasks for natural language processing systems (Grishman 1994). A PCDC system must be able to use the information existing in the corpus in order to assign to each person name mention (PNM) a piece of context relevant for coreference. In many cases, the contextual information relevant for coreference is very scarce or embedded in semantic and ontological deep inferences, which are difficult to program, anyway.

Unlike in other disambiguation tasks, like word sense disambiguation for instance, where the distribution of relevant contexts is mainly regulated by strong syntactic rules, in PCDC the relevance of contexts is a matter of interdependency. To exemplify, consider the name "John Smith" and an organization, say "U.N.". The context "works for U.N." is a relevant coreference context for "John Smith" if there is just one person named John Smith working for U.N.; if there are two or more John Smiths working for U.N., then "works for U.N." is no longer a relevant context for coreference. For the PCDC task, the relevance of the context depends to a great extent on the diversity of the corpus itself, rather than on the specific relationship that exists between "John Smith" and "works for U.N.".

Valid coreference can be realized when a large amount of information is available. However, the requirement that only contextually provable coreferences be realized is too strong; the required relevant context is not actually explicitly found in the text in at least 60% of the times (Popescu 2007).

This paper presents a statistical technique developed to give a PCDC system more information regarding the probability of a correct coreference, without performing deep semantic and ontological analyses. If a PCDC system knows that the prior probability for two PNMs to corefer is high, then the amount of contextual evidence required can be lowered and vice-versa. Our goal is to precisely define a statistical model in which the prior coreference probabilities can be computed and, consequently, to design a PCDC system that dynamically revises the context relevance accordingly.

We review the PCDC literature relevant for our purposes, present the statistical model and show the preliminary results. The paper ends with the Conclusion and Further Research section.

2 Related Work

In a classical paper (Bagga 1998), a PCDC system based on the vector space model (VSM) is proposed. While there are many advantages in representing the context as vectors on which a similarity function is applied, it has been shown that there are

Proceedings of NAACL HLT 2009: Short Papers, pages 153–156,
Boulder, Colorado, June 2009.

inherent limitations associated with the vectorial model (Popescu 2008). These problems, related to the density in the vectorial space (superposition) and to the discriminative power of the similarity power (masking), become visible as more cases are considered. (Gooi, 2004), testing the system on many names, empirically observes the variance in the results obtained by the same PCDC system. Indeed, considering just the sentence level context, which is a strong requirement for establishing coreference, a PCDC system obtains a good score for "John Smith". This is because the probability of coreference of any two "John Smith" mentions is low. But, as the relevant context is often outside the sentence containing the mention, for other types of names the same system is not accurate. If it considers, for instance, "Barack Obama", the same system obtains a very low recall, as the probability of any two "Barack Obama" mentions to corefer is very high. Without further adjustments, a vectorial model cannot resolve the problem of considering too much or too little contextual evidence in order to obtain a good precision for "John Smith" and simultaneously a good recall for "Barack Obama".

The relationship between the prior probabilities and the accuracy of a system is also empirically noted in (Pederson 2005). In their experiment, the authors note that having in the input of the system the correct number of persons carrying the same name is likely to hurt the results of a system based on bigrams. This happens because the amount of context is statically considered. The variance in the results obtained by a PCDC system has been noted also in (Lefever 2007, Popescu 2007).

In order to improve the performances of PCDC systems based on VSM, some authors have focused on methods that allow a better analysis of the context (Ng 2007) combined with a cascade clustering technique (Wei 2006), or have relied on advanced clustering techniques (Chen 2006).

The technique we present in the next section is complementary to these approaches. We propose a statistical model designed to offer to the PCDC systems information regarding the distribution of PNMs in the corpus. This information is used to reduce the contextual data variation and to attain a good balance between precision and recall.

3 Name Perplexity Classes

The amount of contextual information required for the coreference of two or more PNMs depends on several factors. Our working hypothesis is that we can compute a prior probability of coreference for each name and use this probability to control the amount of contextual evidence required. Let us recall the "John Smith" and "Barack Obama" example from the previous section. Both "John" and "Smith" are American common first and last names. The chance that many different persons carry this name is high. On the other hand, as both "Barack" and "Obama" are rare American first and last names respectively, almost surely many mentions of this name refer only to one person. The argument above does not depend on the context, but just on the prior estimation of the usage of those names. Computing an estimation of a name's frequency class, we may decrease or increase the amount of contextual evidence needed accordingly.

To each one-token name we associate the number of different tokens with which it forms a PNM in the corpus. For example, for "John" we can have the set "Smith", "F. Kennedy", "Travolta" etc. We call this number the perplexity of a one-token name. The perplexity gives a direct estimation of the ambiguity of a name in the corpus. In Table 1 we present the relationship between the number of occurrences (in intervals, in the first column) and the average perplexity (second column). The figures reported here, as well as those in the next Section, come from the investigation of the Adige500k, an Italian news corpus (Magnini 2006).

occurrences (interval)	average perplexity
1-5	4.13
6-20	8.34
21-100	17.44
101-1,000	68.54
1,000-5,000	683.95
5,000-31,091	478.23

Table 1. Average perplexity one-token names

We divide the class of one-token names in 5 categories according to their perplexity: very low, low, medium, high and very high. It is useful to keep separate the first and the last names. It has been shown that the average perplexity is three times lower for last names than for first names

(Popescu 2007). Therefore, the first and last names perplexities play different roles in establishing the prior probability of coreference. The perplexity class of two-token names is computed using the following heuristics: the perplexity class of two-token names is the average class of the perplexity of the one-token names composing it. If the perplexity classes of the one-token names are the same, then the perplexity of the whole name is one class less (if possible).

The perplexity classes represent a partition of the name population; each name belongs to one and only one class. In establishing the border between two consecutive perplexity classes, we want to maximize the confidence that inside each stratum the prior coreference probability has a low variance.

The relationship between the perplexity classes and the prior coreference probability is straightforward. The lower the perplexity, the greater the coreference probability, and, therefore, the lower the amount of relevant context required for coreference.

In order to decide the percentage of the name population that goes into each of the perplexity classes, we use a distributional free statistics method. In this way we can compute the confidence of the prior conference probability estimates.

We introduce two random variables: X, a random variable defined over the name population and Y, which represents the number of different persons carrying the same name. Let $X_1,\ldots,X_n$ be a random sample of names from one perplexity class, and let $Y_1,\ldots,Y_n$ be the corresponding values denoting the number of persons that carry the names $X_1,\ldots,X_n$. The indices have been chosen such that $Y_1\ldots,Y_n$ is an ordered statistics: $Y_1 \leq Y_2 \leq \ldots \leq Y_n$. Let F be the distribution function of Y. And let p be a given probability. If $F(Y_j)-F(Y_i) \geq p$, then at least 100p percent of the probability distribution is between Y_i and Y_j; it means that

$$\gamma = P[F(Y_j)-F(Y_i)] \geq p \quad (1)$$

is the probability that the interval (Y_i, Y_j) contains 100p percent of the Y values.

In our case, γ is the confidence of the estimation that 100p percent of names from a certain perplexity class have the expected prior coreference probability in a given interval.

The γ probability is computed with the formula:

$$\gamma = P(F(Y_j) - F(Y_i) < p) = 1-\int_0^p \Gamma(n+1)/(\Gamma(j-i))\,\Gamma(n-j+i+1)x^{j-i-1}(1-x)^{n-j+i}dx \quad (2)$$

where Γ is the extension of the factorial function, $\Gamma(x) = \int_0^\infty t^{x-1}e^{-t}dt$.

In practice, we start with an interval that represents the prior coreference probability desired for that perplexity class. For example we want to be γ = 80% sure that p = 90% of the two-token names in the "very low" perplexity class are names carried by a maximum of 2 persons. We choose a random sample of two-token names from that perplexity class, the size of the random sample being determined by γ and p – see equation (2). If the random sample satisfies (1) then we have the desired perplexity class. If not, the one-token names that have the highest perplexity and were considered "very low" are excluded – they are assigned to the next perplexity class - and the computation is re made.

In a preliminary experiment, using a sample of 25 two-token names from a part of the Adige500k corpus spanning two years, we have obtained the perplexity classes listed in Tables 2 and 3. In Adige 500k there are 106, 192 different one-token names, which combine into 429, 251 different two-token names and 36, 773 three-token names.

perplexity class	percentage
very high	5.3%
High	8.7%
Medium	20.9%
Low	27.6%
very low	37.5%

Table 2. First Name perplexity classes

perplexity class	percentage
very high	1.8%
High	3.36%
Medium	17.51%
Low	20.31%
very low	57.02%

Table 3. Last Name perplexity classes

The perplexity class of two-token names is computed as specified in the first paragraph of this page. In approximately 60% of the cases, a two-token name has a "low", or "very low" perplexity class. If a PCDC system computes the context

similarity based on words with special properties or on named entities, in general at least four similarities must be detected between two contexts in order to have a safe coreference. Our preliminary results show that coreferring on the basis of just one special word and one named entity for those names in "low" or "very low" does not lose more than 1,5% in precision, while it gains up to 40% in recall for these cases. On the other hand, for "very high" perplexity two-token names we were able to increase precision by requiring a stronger similarity between contexts.

The gain of using prior coreference probabilities determined by the perplexity classes is important, especially for those names that are situated at the extreme: "very low" perplexity with a big number of occurrences and "very high" with a small number of occurrences. These cases establish the interval for the amount of contextual similarity required for coreference.

However, the problematic cases remain when the perplexity class is "very high" and the number of occurrences is very big.

4 Conclusion and Further Research

We have presented a distributional free statistical method to design a name perplexity system, such that each perplexity class maximizes the number of names for which the prior coreference belongs to the same interval. This information helps the PCDC systems to lower/increase adequately the amount of contextual evidence required for coreference.

In our preliminary experiment we have observed that we can adequately reduce the amount of contextual evidence required for the coreference of "low" and "very low" perplexity class. For the top perplexity class names the requirement for extra contextual evidence has increased the precision.

The approach presented here is effective in dealing with the problems raised by using a similarity metrics on contextual vectors. It gives a direct way of identifying the most problematic cases for coreference. Solving these cases represents our first objective for the future.

We plan to increase the number of cases considered in the sample required to delimit the perplexity classes. The equation (2) may be developed further in order to obtain exactly the number of required cases for each perplexity class.

References

A. Bagga, B. Baldwin.1998. *Entity-based Cross-Document Co-referencing using the Vector Space Model*, In Proceedings ACL.

J. Chen, D. Ji, C. Tan, Z. Niu.2006. *Unsupervised Relation Disambiguation Using Spectral Clustering,* In Proceedings of COLING

C. Gooi, J. Allan.2004. *Cross-Document Coreference on a Large Scale Corpus*, in Proceeding ACL.

R. Grishman.1994. *Whither Written Language Evaluation?* In proceedings Human Language Technology Workshop, 120-125. San Mateor.

E. Lefever, V. Hoste, F. Timur.2007. *AUG: A Combined Classification and Clustering Approach for Web People Disambiguation*, In Proceedings of SemEval

B. Magnini, M. Speranza, M. Negri, L. Romano, R. Sprugnoli. 2006.I-CAB – the Italian Content Annotation Bank. LREC 2006

V., Ng.2007. *Shallow Semantics for Coreference Resolution,* In Proceedings of IJCAI

T. Pedersen, A. Purandare, A. Kulkarni. 2005. *Name Discrimination by Clustering Similar Contexts*, in Proceeding of CICLING

O. Popescu, C. Girardi, 2008, *Improving Cross Document Coreference*, in Proceedings of JADT

O. Popescu, B. Magnini.2007, *Inferring Coreference among Person Names in a Large Corpus of News Collection*, in Proceedings of AIIA

O. Popescu, B. Magnini.2007. *Irst-bp: WePS using Named Entities*, In Proceedings of SEMEVAL

O. Popescu, M. Magnini, L. Serafini, A. Tamilin, M. Speranza.2006. *From Mention to Ontology: a Pilot Study*, in Proceedings of SWAP

Y. Wei, M. Lin, H. Chen.2006. *Name Disambiguation in Person Information Mining,* In Proceedings of IEEE

Answer Credibility: A Language Modeling Approach to Answer Validation

Protima Banerjee Hyoil Han
College of Information Science and Technology
Drexel University
Philadelphia, PA 19104
pb66@drexel.edu, hyoil.han@acm.org

Abstract

Answer Validation is a topic of significant interest within the Question Answering community. In this paper, we propose the use of language modeling methodologies for Answer Validation, using corpus-based methods that do not require the use of external sources. Specifically, we propose a model for Answer Credibility which quantifies the reliability of a source document that contains a candidate answer and the Question's Context Model.

1 Introduction

In recent years, Answer Validation has become a topic of significant interest within the Question Answering community. In the general case, one can describe Answer Validation as the process that decides whether a Question is correctly answered by an Answer according to a given segment of supporting Text. Magnini et al. (Magnini, 2002) presents an approach to Answer Validation that uses redundant information sources on the Web; they propose that the number of Web documents in which the question and the answer co-occurred can serve as an indicator of answer validity. Other recent approaches to Answer Validation Exercise in the Cross-Language Evaluation Forum (CLEF) (Peters, 2008) make use of textual entailment methodologies for the purposes of Answer Validation.

In this paper, we propose the use of language modeling methodologies for Answer Validation, using corpus-based methods that do not require the use of external sources. Specifically, we propose the development of an Answer Credibility score which quantifies reliability of a source document that contains a candidate answer with respect to the Question's Context Model. Unlike many textual entailment methods, our methodology has the advantage of being applicable to question types for which hypothesis generation is not easily accomplished.

The remainder of this paper describes our work in progress, including our model for Answer Credibility, our experiments and results to date, and future work.

2 Answer Credibility

Credibility has been extensively studied in the field of information science (Metzger, 2002). Credibility in the computational sciences has been characterized as being synonymous with believability, and has been broken down into the dimensions of trustworthiness and expertise.

Our mathematical model of Answer Credibility attempts to quantify the reliability of a source using the semantic Question Context. The semantic Question Context is built using the Aspect-Based Relevance Language Model that was presented in (Banerjee, 2008) and (Banerjee, 2009). This model builds upon the Relevance Based Language Model (Lavrenko, 2001) and Probabilisitic Latent Semantic Analysis (PLSA) (Hofmann, 1999) to provide a mechanism for relating sense disambiguated Concept Terms (CT) to a query by their likelihood of relevance.

The Aspect-Based Relevance Language Model assumes that for every question there exists an un-

Proceedings of NAACL HLT 2009: Short Papers, pages 157–160,
Boulder, Colorado, June 2009.

derlying relevance model R, which is assigned probabilities *P(z|R)* where *z* is a latent aspect of the information need, as defined by PLSA. Thus, we can obtain a distribution of aspects according to their likelihood of relevancy to the user's information need. By considering terms from the aspects that have the highest likelihood of relevance (eg. highest *P(z|R)* values), we can build a distribution that models a semantic Question Context.

We define Answer Credibility to be a similarity measure between the Question Context (QC) and the source document from which the answer was derived. We consider the Question Context to be a document, which has a corresponding document language model. We then use the well-known Kullback-Leibler divergence method (Lafferty, 2001) to compute the similarity between the Question Context document model and the document model for a document containing a candidate answer:

$$AnswerCredibility = \sum_{w \in CT} P(w \mid QC) \log \frac{P(w \mid QC)}{P(w \mid d)}$$

Here, *P(w|QC)* is the language model of the Question Context, *P(w|d)* is the language model o the document containing the candidate answer. To insert this model into the Answer Validation process, we propose an interpolation technique that modulates the answer score during the process using Answer Credibility.

3 Experimental Setup

The experimental methodology we used is shown as a block diagram in Figure 1. To validate our approach, we used the set of all factoid questions from the Text Retrieval Conference (TREC) 2006 Question Answering Track (Voorhees, 2006).

The OpenEphyra Question Answering testbed (Schlaefer, 2006) was then used as the framework for our Answer Credibility implementation. OpenEphyra uses a baseline Answer Validation mechanism which uses documents retrieved using Yahoo! search to support candidate answers found in retrieved passages. In our experiments, we constructed the Question Context according to the methodology described in (Banerjee, 2008). Our experiments used the Lemur Language Modeling toolkit (Strohman, 2005) and the Indri search engine (Ogilvie, 2001) to construct the Question Context and document language models.

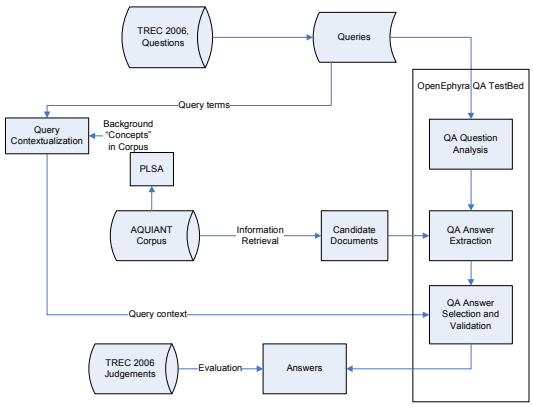


Figure 1: Experiment Methodology

We then inserted an Answer Credibility filter into the OpenEphyra processing pipeline which modulates the OpenEphyra answer score according to the following formula:

$$score' = (1-\lambda) * score + \lambda * AnswerCredibility$$

Here score is the original OpenEphyra answer score and score' is the modulated answer score. In this model, λ is an interpolation constant which we set using the average of the *P(z|R)* values for those aspects that are included in the Question Context.

For the purposes of evaluating the effectiveness of our theoretical model, we use the accuracy and Mean Reciprocal Rank (MRR) metrics (Voorhees, 2005).

4 Results

We compare the results of the baseline OpenEphyra Answer Validation approach against the results after our Answer Credibility processing has been included as a part of the OpenEphyra pipeline. Our results are presented in Table 1 and Table 2.

To facilitate interpretation of our results, we subdivided the set of factoid questions into categories by their question words, following the example of (Murdock, 2006). The light grey shaded cells in both tables indicate categories for which improvements were observed after our Answer Credibility model was applied. The dark grey shaded cells in both tables indicate categories for which no change was observed. The paired Wilcoxon signed rank

test was used to measure significance in improvements for MRR; the shaded cells in Table 2 indicate results for which the results were significant ($p<0.05$). Due to the binary results for accuracy at the question level (eg. a question is either correct or incorrect), the Wilcoxon test was found to be inappropriate for measuring statistical significance in accuracy.

Table 1: Average MRR of Baseline vs. Baseline Including Answer Credibility

Question Category	Question Count	Baseline MRR	Baseline + Answer Credibility MRR
How	20	0.33	0.28
how many	58	0.21	0.16
how much	6	0.08	0.02
in what	47	0.68	0.60
What	114	0.30	0.33
what is	28	0.26	0.26
When	29	0.30	0.19
Where	23	0.37	0.37
where is	6	0.40	0.40
Which	17	0.38	0.26
Who	17	0.51	0.63
who is	14	0.60	0.74
who was	24	0.43	0.55

Table 2: Average Accuracy of Baseline vs. Baseline Including Answer Credibility

Question Category	Question Count	Baseline Accuracy	Baseline + Answer Credibility Accuracy
How	20	0.25	0.20
how many	58	0.12	0.07
how much	6	0.00	0.00
in what	47	0.64	0.55
What	114	0.23	0.28
what is	28	0.18	0.18
When	29	0.21	0.10
Where	23	0.30	0.30
where is	6	0.33	0.33
Which	17	0.29	0.18
Who	17	0.47	0.59
who is	14	0.57	0.71
who was	24	0.38	0.50

Our results show the following:

- A 5% improvement in accuracy over the baseline for "what"-type questions.
- An overall improvement of 13% in accuracy for "who"-type questions, which include the "who," "who is" and "who was" categories
- A 9% improvements in MRR for "what" type questions
- An overall improvement of 25% in MRR for "who"-type questions, which include the "who," "who is" and "who was" categories
- Overall, 7 out of 13 categories (58%) performed at the same level or better than the baseline

5 Discussion

In this section, we examine some examples of questions that showed improvement to better understand and interpret our results.

First, we examine a "who" type question which was not correctly answered by the baseline system, but which was correctly answered after including Answer Credibility. For the question "Who is the host of the Daily Show?" the baseline system correctly determined the answer was "Jon Stewart" but incorrectly identified the document that this answer was derived from. For this question, the Question Context included the terms "stewart," "comedy," "television," "news," and "kilborn." (Craig Kilborn was the host of Daily Show until 1999, which makes his name a logical candidate for inclusion in the Question Context since the AQUAINT corpus spans 1996-2000). In this case, the correct document that the answer was derived from was actually ranked third in the list. The Answer Credibility filter was able to correctly increase the answer score of that document so that it was ranked as the most reliable source for the answer and chosen as the correct final result.

Next, we consider a case where the correct answer was ranked at a lower position in the answer list in the baseline results and correctly raised higher, though not to the top rank, after the application of our Answer Credibility filter. For the question "What position did Janet Reno assume in 1993?" the correct answer ("attorney general") was ranked 5 in the list in the baseline results. However, in this case the score associated with the answer was lower than the top-ranked answer by an order of magnitude. The Question Context for this question included the terms "miami," "elian," "gonzales," "boy," "attorney" and "justice." After the application of our Answer Credibility filter, the score and rank of the correct answer did increase (which con-

tributed to an increase in MRR), but the increase was not enough to overshoot the original top-ranked answer.

Categories for which the Answer Credibility had negative effect included "how much" and "how many" questions. For these question types, the correct answer or correct document was frequently not present in the answer list. In this case, the Answer Credibility filter had no opportunity to increase the rank of correct answers or correct documents in the answer list. This same reasoning also limits our applicability to questions that require a date in response.

Finally, it is important to note here that the very nature of news data makes our methodology applicable to some categories of questions more than others. Since our methodology relies on the ability to derive semantic relationships via a statistical examination of text, it performs best on those questions for which some amount of supporting information is available.

6 Conclusions and Future Work

In conclusion, we have presented a work in progress that uses statistical language modeling methods to create a novel measure called Answer Credibility for the purpose of Answer Validation. Our results show performance increases in both accuracy and MRR for "what" and "who" type questions when Answer Credibility is included as a part of the Answer Validation process. Our goals for the future include further development of the Answer Credibility model to include not only terms from a Question Context, but terms that can be deduced to be in an Answer Context.

References

Banerjee, P., Han, H. 2008. "Incorporation of Corpus-Specific Semantic Information into Question Answering Context," CIKM 2008 - Ontologies and Information Systems for the Semantic Web Workshop, Napa Valley, CA.

Banerjee, P., Han, H 2009. "Modeling Semantic Question Context for Question Answering," *To appear in FLAIRS 2009*.

Hofmann, T. 1999. "Probabilistic latent semantic indexing," Proceedings of the 22nd Annual International SIGIR.

Lafferty, J. and Zhai, C. 2001. "Document language models, query models, and risk minimization for information retrieval," in Proceedings of the 24th Annual International ACM SIGIR, New Orleans, Louisiana: pp. 111-119.

Lavrenko, V. and Croft, W. B. 2001. "Relevance based language models," Proceedings of the 24th annual international ACM SIGIR, pp. 120-127.

Magnini, B., Negri, M., Prevete, R. Tanev, H. 2002. "Is It the Right Answer? Exploiting Web Redundancy for Answer Validation," in Association for Computational Lingustistics (ACL) 2002, Philadelphia, PA, pp. 425-432.

Metzger, M. 2007. "Making sense of credibility on the Web: Models for evaluating online information and recommendations for future research," Journal of the American Society of Information Science and Technology (JASIST), vol. 58, p. 2078.

Murdock, V. 2006. Exploring Sentence Retrieval. VDM Verlag.

Ogilvie, P. and Callan, J. P. 2001. "Experiments Using the Lemur Toolkit," in Online Proceedings of the 2001 Text Retrieval Conference (TREC).

Peters, C. 2008. "What happened in CLEF 2008: Introduction to the Working Notes." http://www.clef-campaign.org/2008/working_notes.

Schlaefer, N., Gieselmann, P., Schaaf, T., & A., W. 2006. A Pattern Learning Approach to Question Answering within the Ephyra Framework, In Proceedings of the Ninth International Conference on Text, Speech and Dialogue (TSD).

Strohman, T., Metzler, D., Turtle, H., and Croft, W. B. 2005. "Indri: A language model-based search engine for complex queries," International Conference on Intelligence Analysis McLean, VA.

Voorhees, E. M. and Harman, D. K. 2005. TREC: Experiment and Evaluation in Information Retrieval (Digital Libraries and Electronic Publishing): The MIT Press.

Voorhees, E. M. 2006. "Overview of the TREC 2006 Question Answering Track," in Online Proceedings of 2006 Text Retrieval Conference (TREC).

Exploiting Named Entity Classes in CCG Surface Realization

Rajakrishnan Rajkumar **Michael White** **Dominic Espinosa**

Department of Linguistics
The Ohio State University
Columbus, OH, USA
{raja,mwhite,espinosa}@ling.osu.edu

Abstract

This paper describes how named entity (NE) classes can be used to improve broad coverage surface realization with the OpenCCG realizer. Our experiments indicate that collapsing certain multi-word NEs and interpolating a language model where NEs are replaced by their class labels yields the largest quality increase, with 4-grams adding a small additional boost. Substantial further benefit is obtained by including class information in the hypertagging (supertagging for realization) component of the system, yielding a state-of-the-art BLEU score of 0.8173 on Section 23 of the CCGbank. A targeted manual evaluation confirms that the BLEU score increase corresponds to a significant rise in fluency.

1 Introduction

Hogan et al. (2007) have recently shown that better handling of named entities (NEs) in broad coverage surface realization with LFG can lead to substantial improvements in BLEU scores. In this paper, we confirm that better NE handling can likewise improve broad coverage surface realization with CCG, even when employing a more restrictive notion of named entities that better matches traditional realization practice. Going beyond Hogan et al. (2007), we additionally show that NE classes can be used to improve realization quality through better language models and better hypertagging (supertagging for realization) models, yielding a state-of-the-art BLEU score of 0.8173 on Section 23 of the CCGbank.

A question addressed neither by Hogan et al. nor anyone else working on broad coverage surface realization recently is whether reported increases in BLEU scores actually correspond to observable improvements in quality. We view this situation as problematic, not only because Callison-Burch et al. (2006) have shown that BLEU does not always rank competing systems in accord with human judgments, but also because surface realization scores are typically much higher than those in MT—where BLEU's performance has been repeatedly assessed—even when using just one reference. Thus, in this paper, we present a targeted manual evaluation confirming that our BLEU score increase corresponds to a significant rise in fluency, a practice we encourage others to adopt.

2 CCG Surface Realization

CCG (Steedman, 2000) is a unification-based categorial grammar formalism defined almost entirely in terms of lexical entries that encode subcategorization as well as syntactic features (e.g. number and agreement). OpenCCG is a parsing/generation library which includes a hybrid symbolic-statistical chart realizer (White, 2006). A vital component of the realizer is the hypertagger (Espinosa et al., 2008), which predicts lexical category assignments using a maxent model trained on contexts within a directed graph structure representing the logical form (LF) input; features and relations in the graph as well as parent child relationships are the main features used to train the model. The realizer takes as input an LF description (see Figure 1 of Espinosa et al., 2008), but here we also

Proceedings of NAACL HLT 2009: Short Papers, pages 161–164,
Boulder, Colorado, June 2009.

use LFs with class information on some elementary predications (e.g. $@_{x:\mathrm{MONEY}}(\mathbf{\$\_10{,}000})$). Chart realization proceeds in iterative beta-best fashion, with a progressively wider hypertagger beam width. If no complete realization is found within the time limit, fragments are greedily assembled. Alternative realizations are ranked using integrated n-gram scoring; n-gram models help in choosing word order and, to a lesser extent, making lexical choices.

3 Collapsing Named Entities

An error analysis of the OpenCCG baseline output reveals that out of 2331 NEs annotated by the BBN corpus, 238 are not realized correctly. For example, multi-word NPs like *Texas Instruments Japan Ltd.* are realized as *Japan Texas Instruments Ltd.*. Inspired by Hogan et al.'s (2007)'s Experiment 1, we decided to use the BBN corpus NE annotation (Weischedel and Brunstein, 2005) to collapse certain classes of NEs. But unlike their experiment where all the NEs annotated by the BBN corpus are collapsed, we chose to collapse into single tokens only NEs whose exact form can be reasonably expected to be specified in the input to the realizer. For example, while some quantificational or comparatives phrases like *more than $ 10,000* are annotated as MONEY in the BBN corpus, in our view only *$_10,000* should be collapsed into an atomic unit, with *more than* handled compositionally according to the semantics assigned to it by the grammar. Thus, after transferring the BBN annotations to the CCGbank corpus, we (partially) collapsed NEs which are CCGbank constituents according to the following rules: (1) completely collapse the PERSON, ORGANIZATION, GPE, WORK OF ART major class type entitites; (2) ignore phrases like *three decades later*, which are annotated as DATE entities; and (3) collapse all phrases with POS tags CD or NNP(S) or lexical items % or *$*, ensuring that all prototypical named entities are collapsed.

4 Exploiting NE Classes

Going beyond Hogan et al. (2007) and collapsing experiments, we also experiment with NE classes in language models and hypertagging models. BBN annotates both major types and subtypes (DATE:AGE, DATE:DATE etc). For all our experiments, we use both of these.

4.1 Class replaced n-gram models

For both the original CCGbank as well as the collapsed corpus, we created language model training data with semantic classes replacing actual words, in order to address data sparsity issues caused by rare words in the same semantic class. For example, in the collapsed corpus, the Section 00 sentence *Pierre_Vinken , 61 years old , will join the board as a nonexecutive director Nov._29 .* becomes *PERSON , DATE:AGE DATE:AGE old , will join the ORG_DESC:OTHER as a nonexecutive PER_DESC DATE:DATE DATE:DATE .* During realization, word forms are generated, but are then replaced by their semantic classes and scored using the semantic class replaced n-gram model, similar to (Oh and Rudnicky, 2002). As the specific words may still matter, the class replaced model is interpolated at the word level with an ordinary, word-based language model, as well as with a factored language model over POS tags and supertags.

4.2 Class features in hypertagging

We also experimented with a hypertagging model trained over the collapsed corpus, where the semantic classes of the elementary lexical predications, along with the class features of their adjacent nodes, are added as features.

5 Evaluation

5.1 Hypertagger evaluation

As Table 2 indicates, the hypertagging model does worse in terms of per-logical predication accuracy & per-whole-graph accuracy on the collapsed corpus. To some extent this is not surprising, as collapsing eliminates many easy tagging cases; however, a full explanation is still under investigation. Note that class information does improve performance somewhat on the collapsed corpus.

5.2 Realizer evaluation

For a both the original CCGbank and the collapsed corpus, we extracted a section 02–21 lexicogrammars and used it to derive LFs for the development and test sections. We used the language models in Table 1 to score realizations and for the

Condition	Expansion
LM	baseline-LM: word 3g+ pos 3g*stag 3g
HT	baseline Hypertagger
LM4	LM with 4g word
LMC	LM with class-rep model interpolated
LM4C	LM with both
HTC	HT with classes on nodes as extra feats

Table 1: Legend for Experimental Conditions

Corpus	Condition	Tags/pred	Pred	Graph
Uncollapsed	HT	1.0	93.56%	39.14%
	HT	1.5	98.28%	78.06%
Partly Collapsed	HT	1.0	92.22%	35.04%
	HTC	1.0	92.89%	38.31%
	HT	1.5	97.87%	73.14%
	HTC	1.5	98.02%	75.30%

Table 2: Hypertagger testing on Section 00 of the uncollapsed corpus (1896 LFs & 38104 predicates) & partially collapsed corpus (1895 LFs & 35370 predicates)

collapsed corpus, we also tried a class-based hypertagging model. Hypertagger β-values were set for each corpus and for each hypertagging model such that the predicted tags per pred was the same at each level. BLEU scores were calculated after removing the underscores between collapsed NEs.

5.3 Results

Our baseline results are much better than those previously reported with OpenCCG in large part due to improved grammar engineering efforts and bug fixing. Table 3 shows development set results which indicate that collapsing appears to improve realization on the whole, as evidenced by the small increase in BLEU scores. The class-replaced word model provides a big boost on the collapsed corpus, from 0.7917 to 0.7993, much more than 4-grams. Adding semantic classes to the hypertagger improves its accuracy and gives us another half BLEU point increase. Standard test set results, reported in Table 4, confirm the overall increase, from 0.7940 to 0.8173.

In analyzing the Section 00 results, we found that with the collapsed corpus, NE errors were reduced from 238 to 99, which explains why the BLEU score increases despite the drop in exact matches and grammatically complete realizations from the baseline. A semi-automatic analysis reveals that most of the corrections involve proper names that are no longer mangled. Correct adjective ordering is also achieved in some cases; for example, *Dutch publishing group* is enforced by the class-replaced models, while all the other models realize this as *publishing Dutch group*. Additionally, the class-replaced model sometimes helps with animacy marking on relative pronouns, as in *Mr. Otero , who ...* instead of *Mr. Otero , which ....* (Note that our input LFs do not directly specify the choice of function words such as case-marking prepositions, relative pronouns and complementizers, and thus class-based scoring can help to select the correct surface word form.)

Corpus	Condition	%Exact	%Complete	BLEU
Uncollapsed (98.6% coverage)	LM+HT	29.27	84.02	0.7900
	LM4+HT	29.14	83.61	0.7899
	LMC+HT	30.64	83.70	0.7937
	LM4C+HT	30.85	83.65	**0.7946**
Partly collapsed (98.6% coverage)	LM+HT	28.28	82.48	0.7917
	LM4+HT	28.68	82.54	0.7929
	LMC+HT	30.74	82.33	0.7993
	LM4C+HT	31.06	82.33	0.7995
	LM4C+HTC	32.01	83.17	**0.8042**

Table 3: Section 00 blind testing results

Condition	%Exact	%Complete	BLEU
LM+HT	29.38	82.53	0.7940
LM4C+HTC	33.74	85.04	**0.8173**

Table 4: Section 23 results: LM+HT baseline on original corpus (97.8% coverage), LM4C+HTC best case on collapsed corpus (94.8% coverage)

5.4 Targeted manual evaluation

While the language models employing NE classes certainly improve some examples, others are made worse, and some are just changed to different, but equally acceptable paraphrases. For this reason, we carried out a targeted manual evaluation to confirm the BLEU results.

5.4.1 Procedure

Along the lines of (Callison-Burch et al., 2006), two native speakers (two of the authors) provided ratings for a random sample of 49 realizations that differed between the baseline and best conditions on the collapsed corpus. Note that the selection procedure excludes exact matches and thus focuses on sentences whose realization quality may be lower on average than in an arbitrary sample. Sentences were rated in the context of the preceding sentence (if any) for both fluency and adequacy in comparison to the original sentence. The judges were not

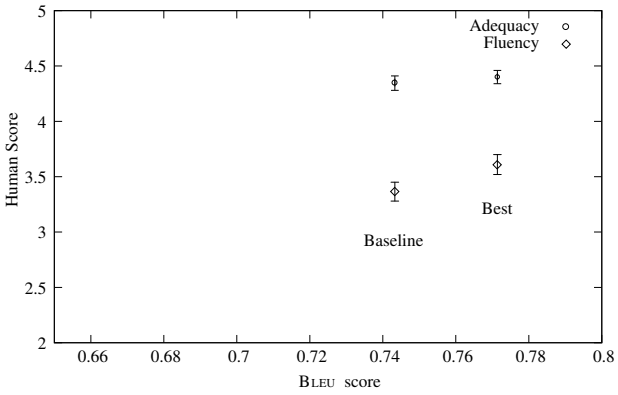


Figure 1: BLEU scores plotted against human judgements of fluency and adequacy

aware of the condition (best/baseline) while doing the rating. Ratings of the two judges were averaged for each item.

5.4.2 Results

In the human evaluation, the best system's mean scores were 4.4 for adequacy and 3.61 for fluency, compared with the baseline's scores of 4.35 and 3.36 respectively. Figure 1 shows these results including the standard error for each measurement, with the BLEU scores for this specific test set. The sample size was sufficient to show that the increase in fluency from 3.36 to 3.61 represented a significant difference (paired t-test, 1-tailed, p = 0.015), while the adequacy scores did not differ significantly.

5.4.3 Brief comparison to related systems

While direct comparisons cannot really be made when inputs vary in their semantic depth and specificity, we observe that our all-sentences BLEU score of 0.8173 exceeds that of Hogan et al. (2007), who report a top score of 0.6882 (though with coverage near 100%). Nakanishi et al. (2005) and Langkilde-Geary (2002) report scores of 0.7733 and 0.7570, respectively, though the former is limited to sentences of length 20 or less, and the latter's coverage is much lower.

6 Conclusion and Future Work

In this paper, we have shown how named entity classes can be used to improve the OpenCCG realizer's language models and hypertagging models, helping to achieve a state-of-the-art BLEU score of 0.8173 on CCGbank Section 23. We have also confirmed the increase in quality through a targeted manual evaluation, a practice we encourage others working on surface realization to adopt. In future work, we plan to investigate the unexpected drop in hypertagger performance on our NE-collapsed corpus, which we conjecture may be resolved by taking advantage of Vadas and Curran's (2008) corrections to the CCGbank's NP structures.

7 Acknowledgements

This work was supported in part by NSF IIS-0812297 and by an allocation of computing time from the Ohio Supercomputer Center. Our thanks also to Josef Van Genabith, the OSU Clippers group and the anonymous reviewers for helpful comments and discussion.

References

Chris Callison-Burch, Miles Osborne, and Philipp Koehn. 2006. Re-evaluating the role of BLEU in machine translation research. In *Proc. EACL*.

Dominic Espinosa, Michael White, and Dennis Mehay. 2008. Hypertagging: Supertagging for surface realization with CCG. In *Proc. ACL-08:HLT*.

Deirdre Hogan, Conor Cafferkey, Aoife Cahill, and Josef van Genabith. 2007. Exploiting multi-word units in history-based probabilistic generation. In *Proc. EMNLP-CoNLL*.

Irene Langkilde-Geary. 2002. An empirical verification of coverage and correctness for a general-purpose sentence generator. In *Proc. INLG-02*.

Hiroko Nakanishi, Yusuke Miyao, and Jun'ichi Tsujii. 2005. Probabilistic methods for disambiguation of an HPSG-based chart generator. In *Proc. IWPT-05*.

Alice H. Oh and Alexander I. Rudnicky. 2002. Stochastic natural language generation for spoken dialog systems. *Computer, Speech & Language*, 16(3/4):387–407.

Mark Steedman. 2000. *The syntactic process*. MIT Press, Cambridge, MA, USA.

David Vadas and James R. Curran. 2008. Parsing noun phrase structure with CCG. In *Proc. ACL-08:HLT*.

Ralph Weischedel and Ada Brunstein. 2005. BBN pronoun coreference and entity type corpus. Technical report, BBN.

Michael White. 2006. Efficient Realization of Coordinate Structures in Combinatory Categorial Grammar. *Research on Language and Computation*, 4(1):39–75.

Search Result Re-ranking by Feedback Control Adjustment for Time-sensitive Query

Ruiqiang Zhang[†] and **Yi Chang**[†] and **Zhaohui Zheng**[†]
Donald Metzler[†] and **Jian-yun Nie**[‡]
[†]Yahoo! Labs, 701 First Avenue, Sunnyvale, CA94089
[‡]University of Montreal, Montreal, Quebec,H3C 3J7, Canada
[†]{ruiqiang,yichang,zhaohui,metzler}@yahoo-inc.com
[‡]nie@iro.umontreal.ca

Abstract

We propose a new method to rank a special category of time-sensitive queries that are year qualified. The method adjusts the retrieval scores of a base ranking function according to time-stamps of web documents so that the freshest documents are ranked higher. Our method, which is based on feedback control theory, uses ranking errors to adjust the search engine behavior. For this purpose, we use a simple but effective method to extract year qualified queries by mining query logs and a time-stamp recognition method that considers titles and urls of web documents. Our method was tested on a commercial search engine. The experiments show that our approach can significantly improve relevance ranking for year qualified queries even if all the existing methods for comparison failed.

1 Introduction

Relevance ranking plays a crucial role in search engines. There are many proposed machine learning based ranking algorithms such as language modeling-based methods (Zhai and Lafferty, 2004), RankSVM (Joachims, 2002), RankBoost (Freund et al., 1998) and GBrank (Zheng et al., 2007). The input to these algorithms is a set of feature vectors extracted from queries and documents. The goal is to find the parameter setting that optimizes some relevance metric given training data. While these machine learning algorithms can improve *average* relevance, they may be ineffctive for certain special cases. Time-sensitive queries are one such special case that machine-learned ranking functions may have a hard time learning, due to the small number of such queries.

Consider the query "sigir" (the name of a conference), which is time sensitive. Table 1 shows two example search result pages for the query, SERP1 and SERP2. The ranking of SERP2 is clearly better than that of SERP1 because the most recent event, "sigir2009", is ranked higher than other years.

query: sigir	
SERP1	url1: http://www.sigir.org url2: http://www.sigir2008.org url3: http://www.sigir2004.org url4: http://www.sigir2009.org url5: http://www.sigir2009.org/schedule
SERP2	url1: http://www.sigir.org url2: http://www.sigir2009.org url3: http://www.sigir2009.org/schedule url4: http://www.sigir2008.org url5: http://www.sigir2004.org

Table 1: Two contrived search engine result pages

Time is an important dimension of relevance in web search, since users tend to prefer recent documents to old documents. At the time of this writing (February 2009), none of the major commercial search engines ranked the homepage for SIGIR 2009 higher than previous SIGIR homepages for the query "sigir". One possible reason for this is that ranking algorithms are typically based on anchor text features, hyperlink induced features, and click-through rate features. However, these features tend to favor old pages more than recent ones. For example, "sigir2008" has more links and clicks than "sigir2009" because "sigir2008" has existed longer time and therefore has been visited more. It is less likely that newer web pages from "sigir2009" can be ranked higher using features that implicitly favor old pages.

However, the fundamental problem is that current approaches have focused on improving general ranking algorithms. Methods for improving ranking of specific types of query like temporal queries are often overlooked.

Aiming to improve ranking results, some methods of re-ranking search results are proposed, such as the work by (Agichtein et al., 2006) and (Teevan et al., 2005).

Proceedings of NAACL HLT 2009: Short Papers, pages 165–168,
Boulder, Colorado, June 2009.

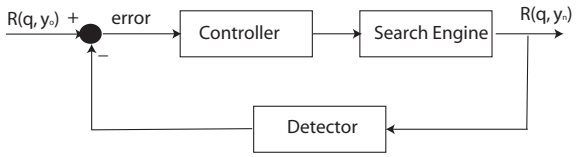


Figure 1: Feedback control for search engine

These work uses user search behavior information or personalization information as features that are integrated into an enhanced ranking model. We propose a novel method of re-ranking search results. This new method is based on feedback control theory, as illustrated in 1.

We make a Detector to monitor search engine (SE) output and compare it with the input, which is the desired search engine ranking. If an error is found, we design the controller that uses the error to adjust the search engine output, such that the search engine output tracks the input. We will detail the algorithm in Section 4.1.

Our method was applied to a special class of time-sensitive query, *year qualified queries* (YQQs). For this category, we found users either attached a year with the query explicitly, like "sigir 2009", or used the query only without a year attached,like "sigir". We call the former explicit YQQs, and the latter implicit YQQs. Using query log analysis, we found these types of queries made up about 10% of the total query volume. We focus exclusively on implicit YQQs by translating the user's implicit intention as the most recent year. Explicit YQQs are less interesting, because the user's temporal intention is clearly specified in the query. Therefore, ranking for these types of queries is relatively straightforward. Throughout the remainder of this paper, we use the "YQQ" to refer to implicit YQQs, unless otherwise stated.

2 Adaptive score adjustment

Our proposed re-ranking model is shown in Eq. 1, as below.

$$
\begin{aligned}
F(q,d) &= \begin{cases} R(q,d) & \texttt{if } q \notin \texttt{YQQ} \\ R(q,d)+Q(q,d) & \texttt{otherwise} \end{cases} \\
Q(q,d) &= \begin{cases} (e(d_o,d_n)+k)e^{\lambda\alpha(q)} & \texttt{if } y(d)=y_n \\ 0 & \texttt{otherwise} \end{cases} \\
e(d_o,d_n) &= R(q,d_o)-R(q,d_n)
\end{aligned}
\tag{1}
$$

This work assumes that a base ranking function is used to rank documents with respect to an incoming query. We denote this base ranking function as $R(q,d)$. This ranking function is conditioned on a query q and a document d. It is assumed to model the relevance between q and d.

Our proposed method is flexible for all YQQ queries. Suppose the current base ranking function gives the results as SERP1 of Table 1. To correct the ranking, we propose making an adjustment to $R(q,d)$.

In Eq. 1, $F(q,d)$ is the final ranking function. If the query is not an YQQ, the base ranking function is used. Otherwise, we propose an adjustment function, $Q(q,d)$, to adjust the base ranking function. $Q(q,d)$ is controlled by the ranking error, $e(d_o,d_n)$, signifying the base function ranking error if the newest web page d_n is ranked lower than the oldest web page d_o. $y(d)$ is the year that the event described by d has occurred or will occur. If y_o and y_n indicate the oldest year and the newest year, then $y(d_o)=y_o, y(d_n)=y_n$. $R(q,d_o)$ and $R(q,d_n)$ are the base ranking function scores for the oldest and the newest documents.

k is a small shift value for direction control. When $k<0$, the newest document is adjusted slightly under the old one. Otherwise, it is adjusted slightly over the old one. Experiments show $k>0$ gave better results. The value of k is determined in training.

$\alpha(q)$ is the confidence score of a YQQ query, meaning the likelihood of a query to be YQQ. The confidence score is bigger if a query is more likely to be YQQ. More details are given in next section. λ is a weighting parameter for adjusting $\alpha(q)$.

The exp function $e^{\lambda\alpha(q)}$ is a weighting to control boosting value. A higher value, confidence α, a larger boosting value, $Q(q,d)$.

Our method can be understood by feedback control theory, as illustrated in Fig. 1. The ideal input is $R(q,y_o)$ representing the desired ranking score for the newest Web page, $R(q,y_n)$. But the search engine real output is $R(q,y_n)$. Because search engine is a dynamic system, its ranking is changing over time. This results in ranking errors, $e(d_o,d_n)=R(q,d_o)-R(q,d_n)$. The function of "Controller" is to design a function to adjust the search engine ranking so that the error approximates to zero, $e(d_o,d_n)=0$. For this work, "Controller" is $Q(q,d)$. "Detector" is a document year-stamp recognizer, which will be described more in the next section. "Detector" is used to detect the newest Web pages and their ranking scores. Fig. 1 is an ideal implementation of our methods. We cannot carry out real-time experiments in this work. Therefore, the calculation of ranking errors was made in offline training.

3 YQQ detection and year-stamp recognition

To implement Eq. 1, we need to find YQQ queries and to identify the year-stamp of web documents.

Our YQQ detection method is simple, efficient, and relies only on having access to a query log with frequency information. First, we extracted all explicit YQQs from

query log. Then, we removed all the years from explicit YQQs. Thus, implicit YQQs are obtained from explicit YQQs. The implicit YQQs are saved in a dictionary. In online test, we match input queries with each of implicit YQQs in the dictionary. If an exact match is found, we regard the input query as YQQ, and apply Eq. 1 to re-rank search results.

After analyzing samples of the extracted YQQs, we group them into three classes. One is recurring-event query, like "sigir", "us open tennis"; the second is newsworthy query, like "steve ballmer", "china foreign reserves"; And the class not belong to any of the above two, like "christmas", "youtube". We found our proposed methods were the most effective for the first category. In Eq. 1, we can use confidence $\alpha(q)$ to distinguish the three categories and their change of ranking as shown in Eq.1, that is defined as below.

$$\alpha(q) = \frac{\sum_y w(q,y)}{\#(q) + \sum_y w(q,y)} \quad (2)$$

where $w(q,y) = \#(q.y)+\#(y.q)$. $\#(q.y)$ denotes the number of times that the base query q is post-qualified with the year y in the query log. Similarly, $\#(y.q)$ is the number of times that q is pre-qualified with the year y. This weight measures how likely q is to be qualified with y, which forms the basis of our mining and analysis. $\#(q)$ is the counts of independent query, without associating with any other terms.

We also need to know the year-stamp $y(d)$ for each web document so that the ranking score of a document is updated if $y(d) = y_n$ is satisfied. We can do this from a few sources such as title, url, anchar text, and extract date from documents that is possible for many news pages. For example, from url of the web page, "www.sigir2009.org", we detect its year-stamp is 2009.

We have also tried to use some machine generated dates. However, in the end we found such dates are inaccurate and cannot be trusted. For example, discovery time is the time when the document was found by the crawler. But a web document may exist several years before a crawler found it. We show the worse effect of using discovery time in the experiments.

4 Experiments

We will describe the implementation methods and experimental results in this section. Our methods include offline dictionary building and online test. In offline training, our first step is to mine YQQs. A commercial search engine company provided us with six months of query logs. We extracted a list of YQQs using Section 3's method. For each of the YQQs, we run the search engine and output the top N results. For each document, we used the method described in Section 3 to recognize the year-stamp and find the oldest and the newest page. If there are multiple urls with the same yearstamp, we choose the first oldest and the first most recent. Next,we calculated the boosting value according to Eq. 1. Each query has a boosting value. For online test, a user's query is matched with each of the YQQs in the dictionary. If an exact match is found, the boosting value will be added to the base ranking score iff the document has the newest yearstamp.

For evaluating our methods, we randomly extracted 600 YQQs from the dictionary. We extracted the top-5 search results for each of queries using the base ranking function and the proposed ranking function. We asked human editors to judge all the scraped results. We used five judgment grades: Perfect, Excellent, Good, Fair, and Bad. Editors were instructed to consider temporal issues when judging. For example, sigir2004 is given a worse grade than sigir2009. To avoid bias, we advised editors to retain relevance as their primary judgment criteria. Our evaluation metric is relative change in DCG, $\%\Delta_{dcg} = \frac{DCG_{proposed} - DCG_{baseline}}{DCG_{baseline}}$, where DCG is the traditional Discounted Cumulative Gain (Jarvelin and Kekalainen, 2002).

4.1 Effect of the proposed boosting method

Our experimental results are shown in Table 2, where we compared our work with the existing methods. While we cannot apply (Li and Croft, 2003)'s approach directly because first, our search engine is not based on language modeling; second, it is impossible to obtain exact timestamp for web pages as (Li and Croft, 2003) did in the track evaluation. However, we tried to simulate (Li and Croft, 2003)'s approach in web search by using the linear integration method exactly as the same as(Li and Croft, 2003) by adding a time-based function with our base ranking function. For the timestamp, we used discovery time in the time-based function. The parameters (λ, α) have the exact same meaning as in (Li and Croft, 2003) but were tuned according to our base ranking function. With regards to the approach by (Diaz and Jones, 2004), we ranked the web pages in decreasing order of discovery time. Our own approaches were tested under options with and without using adaptation. For no adaption, we let the e of Eq.1 equal to 0, meaning no score difference between the oldest document and the newest document was captured, but a constant value was used. It is equivalent to an open loop in Fig.1. For adaption, we used the ranking errors to adjust the base ranking. In the Table we used multiple ks to show the effect of changing k. Using different k can have a big impact on the performance. The best value we found was $k = 0.3$. In this experiment, we let $\alpha(q) = 0$ so that the result responds to k only.

Our approach is significantly better than the existing methods. Both of the two existing methods produced worse results than the baseline, which shows the ap-

Li & Croft	(λ, α)=(0.2,2.0)	-0.5
	(λ, α)=(0.2,4.0)	-1.2
Diaz & Jones		-4.5*
No adaptation ($e = 0$, open loop)	k=0.3	1.2
	k=0.4	0.8
Adaptation (closed loop)	k=0.3	6.6*
	k=0.4	6.2*

Table 2: $\%\Delta_{dcg}$ of proposed method comparing with existing methods.A sign "$*$" indicates statistical significance (p-value<0.05)

λ	0	0.2	0.4	0.6	0.8	1.0
$\%\Delta_{dcg}$	6.6*	7.8*	8.4*	4.5	2.1	-0.2*

Table 3: Effect of confidence as changing λ.

proaches may be inappropriate for Web search. Not surprisingly, using adaption achieved much better results than without using adaption. Thus, these experiments prove the effectiveness of our proposed methods.

Another important parameter in the Eq.1 is the confidence score $\alpha(q)$, which indicates the confidence of query to be YQQ. In Eq. 1, λ is used to adjusting $\alpha(q)$. We observed dcg gain for each different λ. The results are shown in Table 3. The value of λ needs to be tuned for different base ranking functions. A higher λ can hurt performance. In our experiments, the best value of 0.4 gave a 8.4% statistically significant gain in DCG. The $\lambda = 0$ setting means we turn off confidence, which results in lower performance. Thus, using YQQ confidence is effective.

5 Discussions and conclusions

In this paper, we proposed a novel approach to solve YQQ ranking problem, which is a problem that seems to plague most major commercial search engines. Our approach for handling YQQs does not involve any query expansion that adds a year to the query. Instead, keeping the user's query intact, we re-rank search results by adjusting the base ranking function. Our work assumes the intent of YQQs is to find documents about the most recent year. For this reason, we use YQQ confidence to measure the probability of this intent. As our results showed, our proposed method is highly effective. A real example is given in Fig. 2 to show the significant improvement by our method.

Our adaptive methods are not limited to YQQs only. We believe this framework can be applied to any category of queries once a query classification and a score detector have been implemented.

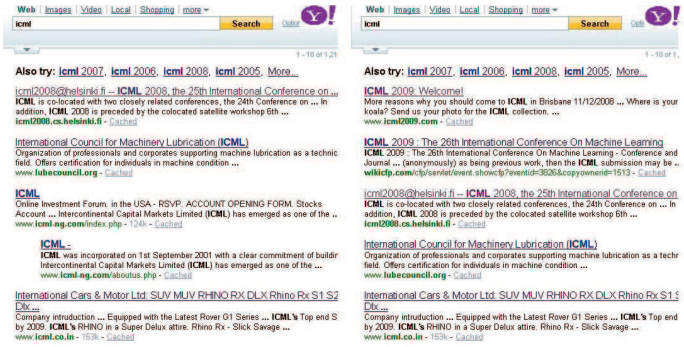


Figure 2: Ranking improvement for query ICML by our method: before re-rank(left) and after(right)

References

Eugene Agichtein, Eric Brill, and Susan Dumais. 2006. Improving web search ranking by incorporating user behavior information. In *SIGIR '06*, pages 19–26.

Fernando Diaz and Rosie Jones. 2004. Using temporal profiles of queries for precision prediction. In *Proc. 27th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, pages 18–24, New York, NY, USA. ACM.

Yoav Freund, Raj D. Iyer, Robert E. Schapire, and Yoram Singer. 1998. An efficient boosting algorithm for combining preferences. In *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*, pages 170–178.

Kalervo Jarvelin and Jaana Kekalainen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20:2002.

Thorsten Joachims. 2002. Optimizing search engines using clickthrough data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142.

Xiaoyan Li and W. Bruce Croft. 2003. Time-based language models. In *Proc. 12th Intl. Conf. on Information and Knowledge Management*, pages 469–475, New York, NY, USA. ACM.

Jaime Teevan, Susan T. Dumais, and Eric Horvitz. 2005. Personalizing search via automated analysis of interests and activities. In *SIGIR '05*, pages 449–456.

Chengxiang Zhai and John Lafferty. 2004. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst.*, 22(2):179–214.

Zhaohui Zheng, Keke Chen, Gordon Sun, and Hongyuan Zha. 2007. A regression framework for learning ranking functions using relative relevance judgments. In *SIGIR '07*, pages 287–294.

A Local Tree Alignment-based Soft Pattern Matching Approach for Information Extraction

Seokhwan Kim, Minwoo Jeong, and Gary Geunbae Lee
Department of Computer Science and Engineering
Pohang University of Science and Technology
San 31, Hyoja-dong, Nam-gu, Pohang, 790-784, Korea
{megaup, stardust, gblee}@postech.ac.kr

Abstract

This paper presents a new soft pattern matching method which aims to improve the recall with minimized precision loss in information extraction tasks. Our approach is based on a local tree alignment algorithm, and an effective strategy for controlling flexibility of the pattern matching will be presented. The experimental results show that the method can significantly improve the information extraction performance.

1 Introduction

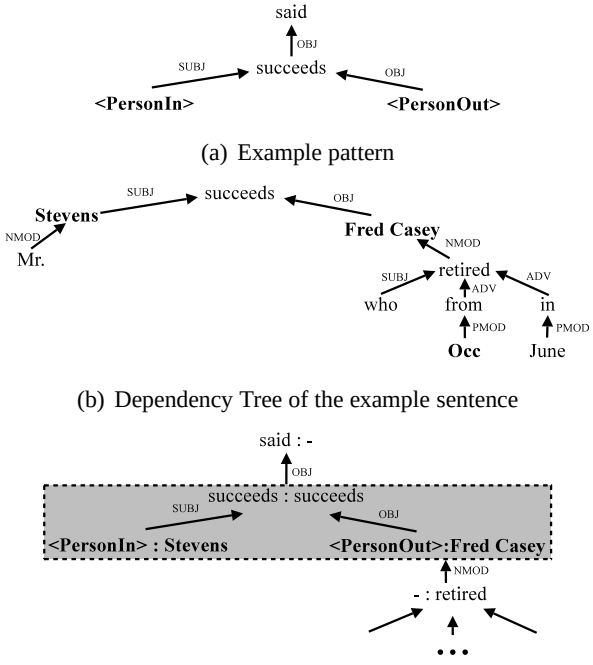
The goal of information extraction (IE) is to extract structured information from unstructured natural language documents. Pattern induction to generate extraction patterns from a number of training instances is one of the most widely applied approaches for IE.

A number of pattern induction approaches have recently been researched based on the dependency analysis (Yangarber, 2003) (Sudo et al., 2001) (Greenwood and Stevenson, 2006) (Sudo et al., 2003). The natural language texts in training instances are parsed by dependency analyzer and converted into dependency trees. Each subtree of a dependency tree is considered as a candidate of extraction patterns. An extraction pattern is generated by selecting the subtree which indicates the dependency relationships of each labeled slot value in the training instance and agrees on the selection criteria defined by each pattern representation model. A number of dependency tree-based pattern representation models have been proposed. The predicate-argument (SVO) model allows subtrees containing only a verb and its direct subject and object as extraction pattern candidates (Yangarber, 2003). The chain model represents extraction patterns as a chain-shaped path from each target slot value to the root node of the dependency tree (Sudo et al., 2001). A couple of chain model patterns sharing the same verb are linked to each other and construct a linked-chain model pattern (Greenwood and Stevenson, 2006). The subtree model considers all subtrees as pattern candidates (Sudo et al., 2003).

Regardless of the applied pattern representation model, the methods have concentrated on extracting only exactly equivalent subtrees of test instances to the extraction patterns, which we call hard pattern matching. While the hard pattern matching policy is helpful to improve the precision of the extracted results, it can cause the low recall problem. In order to tackle this problem, a number of soft pattern matching approaches which aim to improve recall with minimized precision loss have been applied to the linear vector pattern models by introducing a probabilistic model (Xiao et al., 2004) or a sequence alignment algorithm (Kim et al., 2008).

In this paper, we propose an alternative soft pattern matching method for IE based on a local tree alignment algorithm. While other soft pattern matching approaches have been able to handle the matching among linear vector instances with features from tree structures only, our method aims to directly solve the low recall problem of tree-to-tree pattern matching by introducing the local tree alignment algorithm which is widely used in bioinformatics to analyze RNA secondary structures. Moreover,

Proceedings of NAACL HLT 2009: Short Papers, pages 169–172,
Boulder, Colorado, June 2009.



(a) Example pattern

(b) Dependency Tree of the example sentence

(c) Local alignment-based tree pattern matching

Figure 1: An example of local alignment-based tree pattern matching

we present an effective policy for controlling degree of flexibility in the pattern matching by setting the optimal threshold values for each extracted pattern.

2 Methods

The low recall problem of information extraction based on hard pattern matching is caused by lack of flexibility in pattern matching. For example, the tree pattern in Figure 1(a) cannot be matched with the tree in Figure 1(b) by considering only exactly equivalent subtrees, because the first tree has an additional root node 'said' which is not in the second one. However, the matching between two trees can be performed by omitting just a node as shown in Figure 1(c).

In order to improve and control the degree of flexibility in tree pattern matching, we have adopted a local tree alignment approach as the pattern matching method instead of hard pattern matching strategy. The local tree alignment problem is to find the most similar subtree between two trees.

We have adopted the Hochsmann algorithm (Hochsmann et al., 2003) which is a local tree alignment algorithm used in bioinformatics to analyze RNA secondary structures. The goal of the Hochsmann algorithm is to find the local closed forest alignment which maximizes the similarity score for ordered trees. The algorithm can be implemented by a dynamic programming approach which solves a problem based on the previous results of its subproblems. The main problem of Hochsmann algorithm is to compute the similarity score between two subforests according to the defined order from the single node level to the entire tree level. The similarity score is defined based on three tree edit operations which are insertion, deletion, and replacement (Tai, 1979). For each pair of subforests, the maximum similarity score among three edit operations is computed, and the kind and the position of performed edit operations are recorded.

The adaptation of Hochsmann algorithm to the IE problem is performed by redefining the σ-function, the similarity score function between two nodes, as follows:

$$\sigma(v,w) = \begin{cases} 1 & \text{if lnk}(v)\text{=lnk}(w), \\ & \text{and lbl}(v)\text{=lbl}(w), \\ \sigma(\text{p}(w),\text{p}(v)) & \text{if lbl}(v)\text{=<SLOT>}, \\ 0 & \text{otherwise.} \end{cases}$$

where v and w are nodes to be compared, lnk(v) is the link label of v, lbl(v) is the node label of v, and $p(v)$ denotes a parent node of v. While general local tree alignment problems consider only node labels to compute the node-level similarities, our method considers not only node labels, but also link labels to the head node, because the class of link to the head node is important as the node label itself for dependency trees. Moreover, the method should consider the alignment of slot value nodes in the tree patterns for adopting information extraction tasks. If the pattern node v is a kind of slot value nodes, the similarity score between v and w is inherited from parents of both nodes.

After computing for all pairs of subforests, the optimal alignment is obtained by trace-back based on the recorded information of edit operation which maximizes the similarity score for each subforest pair. On the optimal alignment, the target node aligned to a slot value node on the pattern is regarded as an argument candidate of the extraction. Each ex-

traction candidate has its confidence score which is computed from the alignment score, defined as:

$$\text{score}(T_{\text{PTN}}, T_{\text{TGT}}) = \frac{S(T_{\text{PTN}}, T_{\text{TGT}})}{|T_{\text{PTN}}|}$$

where $|T|$ denotes the total number of nodes in tree T and $S(T_1, T_2)$ is the similarity score of both trees computed by Hochsmann algorithm.

Only the extraction candidates with alignment score larger than the given threshold value, θ, are accepted and regarded as extraction results. For the simplest approach, the same threshold value, θ, can be applied to all the patterns. However, we assumed that each pattern has its own optimal threshold value as its own confidence score, which is different from other patterns' threshold values. The optimal threshold value θ_i and the confidence score $conf_i$ for the pattern P_i are defined as:

$$\theta_i = \underset{0.5<\theta\leq 1.0}{\arg\max} \{\text{eval}_{\text{fscore}}(D_{\text{train}}, P_i, \theta)\}$$

$$conf_i = \max_{0.5<\theta\leq 1.0} \{\text{eval}_{\text{fscore}}(D_{\text{train}}, P_i, \theta)\}$$

where $\text{eval}_{\text{fscore}}(D, P, \theta)$ is the evaluation result in F-score of the extraction for the data set D using the pattern P with the threshold value θ. For each pattern, the threshold value which maximizes the evaluation result in F-score for the training data set and the maximum evaluation result in F-score are assigned as the optimal threshold value and the confidence score for the pattern respectively.

3 Experiment

In order to evaluate the effectiveness of our method, we performed an experiment for the scenario template extraction task on the management succession domain in MUC-6. The task aims to extract scenario template instances which consist of person-in, person-out, position, organization slot values from news articles about management succession events. We used a modified version of the MUC-6 corpus including 599 training documents and 100 test documents described by Soderland (1999). While the scenario templates on the original MUC-6 corpus are labeled on each document, this version has scenario templates for each sentence.

All the sentences in both training and test documents were converted into dependency trees by Berkeley Parser[1] and LTH Constituent-to-Dependency Conversion Tool[2]. From the dependency trees and scenario templates on the training data, we constructed pattern candidate sets for four types of pattern representation models which are SVO, chain, linked-chain, and subtree models. For each pattern candidate, corresponding confidence score and optimal threshold value were computed.

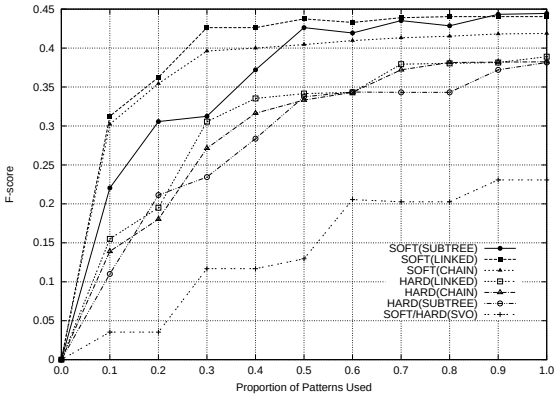


Figure 2: Comparison of soft pattern matching strategy with the hard pattern matching

The pattern candidates for each pattern representation model were arranged in descending order of confidence score. According to the arranged order, each pattern was matched with test documents and the extracted results were accumulated. Extracted templates for test documents are evaluated by comparing with the answer templates on the test corpus.

The curves in Figure 2 show the relative performance of the pattern matching strategies for each pattern representation model. The results suggest that soft pattern matching strategy with optimal threshold values requires less number of patterns for the performance saturation than the hard pattern matching strategy for all pattern models except the SVO model. For the SVO model, the result of soft pattern matching strategy is equivalent to that of hard pattern matching strategy. It is because most of patterns represented in SVO model are relatively shorter than those represented in other models.

In order to evaluate the flexibility controlling strategy, we compared the result of optimally determined threshold values with the cases of using

[1] http://nlp.cs.berkeley.edu/pages/Parsing.html

[2] http://nlp.cs.lth.se/pennconverter/

θ	SVO			Chain			Linked-Chain			Subtree		
	P	R	F	P	R	F	P	R	F	P	R	F
0.7	32.1	18.0	23.1	27.6	55.0	36.8	26.8	57.0	36.4	26.6	58.0	36.5
0.8	32.1	18.0	23.1	43.8	35.0	38.8	43.4	36.0	39.3	44.7	34.0	38.6
0.9	32.1	18.0	23.1	45.2	33.0	38.1	43.8	35.0	38.9	45.2	33.0	38.2
1.0 (hard)	32.1	18.0	23.1	45.2	33.0	38.1	43.8	35.0	38.9	45.2	33.0	38.2
optimal	32.1	18.0	23.1	36.0	49.0	**41.5**	40.7	48.0	**44.0**	43.0	46.0	**44.4**

Table 1: Experimental Results

various fixed threshold values. Table 1 represents the final results for all pattern representation models and threshold values. For the SVO model, all the results are equivalent regardless of the threshold strategy because of extremely short length of the patterns. For the other pattern models, precisions are increased and recalls are decreased by increasing the threshold. The maximum performances in F-score are achieved by our optimal threshold determining strategy for all pattern representation models. The experimental results of our method show the better recall than the cases of hard pattern matching and controlled precision than the cases of extremely soft pattern matching.

4 Conclusion

We presented a local tree alignment based soft pattern matching approach for information extraction. The softness of the pattern matching method is controlled by the threshold value of the alignment score. The optimal threshold values are determined by self-evaluation on the training data. Experimental results indicate that our soft pattern matching approach is helpful to improve the pattern coverage and our threshold learning strategy is effective to reduce the precision loss followed by the soft pattern matching method.

The goal of local tree alignment algorithm is to measure the structural similarity between two trees. It is similar to the kernel functions in the tree kernel method which is another widely applied approach to solve the IE problems. In the future, we plan to incorporate our alignment-based soft pattern matching method into the tree kernel method for IE.

Acknowledgments

This work was supported by the Korea Science and Engineering Foundation(KOSEF) grant funded by the Korea government(MEST) (No. R01-2008-000-20651-0)

References

Mark A. Greenwood and Mark Stevenson. 2006. Improving semi-supervised acquisition of relation extraction patterns. In *Proceedings of Workshop on Information Extraction Beyond The Document*, pp. 29–35.

Matthias Hochsmann, Thomas Toller, Robert Giegerich, and Stefan Kurtz. 2003. Local similarity in rna secondary structures. In *Proceedings of the IEEE Computer Society Bioinformatics Conference* , pp. 159–68.

Seokhwan Kim, Minwoo Jeong, and Gary Geunbae Lee. 2008. An alignment-based pattern representation model for information extraction. In *Proceedings of the ACM SIGIR '08*, pp. 875–876.

Stephen Soderland. 1999. Learning information extraction rules for semi-structured and free text. *Machine Learning*, 34(1):233–272.

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. 2001. Automatic pattern acquisition for japanese information extraction. In *Proceedings of the first international conference on Human language technology research*, pp. 1–7.

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. 2003. An improved extraction pattern representation model for automatic ie pattern acquisition. In *Proceedings of the ACL '03*, pp. 224–231.

Kuo-Chung Tai. 1979. The tree-to-tree correction problem. *Journal of the ACM (JACM)*, 26(3):422–433.

Jing Xiao, Tat-Seng Chua, and Hang Cui. 2004. Cascading use of soft and hard matching pattern rules for weakly supervised information extraction. In *Proceedings of COLING '04*, pp. 542–548.

Roman Yangarber. 2003. Counter-training in discovery of semantic patterns. In *Proceedings of the ACL '03*, pp. 343–350.

Classifying Factored Genres with Part-of-Speech Histograms

S. Feldman, M. Marin, J. Medero, and M. Ostendorf
Dept. of Electrical Engineering
University of Washington, Seattle, Washington 98195
{sergeyf,amarin,jmedero,ostendor}@u.washington.edu

Abstract

This work addresses the problem of genre classification of text and speech transcripts, with the goal of handling genres not seen in training. Two frameworks employing different statistics on word/POS histograms with a PCA transform are examined: a single model for each genre and a factored representation of genre. The impact of the two frameworks on the classification of training-matched and new genres is discussed. Results show that the factored models allow for a finer-grained representation of genre and can more accurately characterize genres not seen in training.

1 Introduction

With increasing quantities of text and transcribed speech available online, the ability to categorize documents based on characteristics beyond topic becomes ever more important. In particular, the genre of a document – whether it is a news report or an editorial, a speech transcript or a weblog – may be relevant for many human tasks. For example, one might want to find "speeches on ethanol" or "weblog entries on Fannie Mae, sorted by most formal first." Genre classification is also of growing importance for human language technologies, such as speech recognition, parsing, and translation, because of the potentially large differences in language associated with genre. Researchers find that genre-dependent models lead to improved performance on these tasks, e.g. (Wang, 2008). Since text harvested from the web is increasingly used to address problems due to sparse training data, genre classification can be useful for sampling such text sources to obtain a better match to the target domain for offline language model training. Prior work on genre-dependent web text filtering for language modeling relied on standard search engine methods, designing queries based on frequent n-grams in the domain, e.g. (Bulyko et al., 2007). However, as the variety of genres online has grown, this method has become less reliable. This work addresses explicit genre classification, with the assumption that genre representation in the training data is incomplete.

In prior work on genre classification, an important question has been the definition of "genre." For many studies, genre has been associated with categories of text, such as research article, novel, news report, editorial, advertisement, etc. In particular, several studies use classes identified in the Brown corpus or the British National Corpus. Spoken genres, including conversation, interview, debate, and planned speech are considered in (Santini, 2004). Examples of spoken and written genres, represented in several corpora available from the Linguistics Data Consortium, are explored in (Feldman et al., 2009). Yet another study focuses on internet-specific document types, including different types of home pages (personal, public, commercial), bulletin boards, and link lists (Lim et al., 2004). A limitation of all of this work is that only a small, fixed set of different genres are explored, with performance assessed on matched data. In this paper, we assess classification results of texts that come from new genres, as well as those matching the training set.

In addressing new genres, we have two main contributions: new features and factored coding.

Proceedings of NAACL HLT 2009: Short Papers, pages 173–176,
Boulder, Colorado, June 2009.

The standard features for genre classification models include words, part-of-speech (POS) tags, and punctuation (Kessler et al., 1997; Stamatatos et al., 2000; Lee and Myaeng, 2002; Biber, 1993), but constituent-based syntactic categories have also been explored (Karlgren and Cutting, 1994). (Feldman et al., 2009) used mixed word and POS histogram mean and variance as features for genre classification. In this work, we augment those histogram statistics with higher-order ones, as well as add new word features aimed at capturing online genres. Further, we propose a factored genre model, and demonstrate its effect on genre classification of out-of-domain documents.

2 Methods

2.1 Corpora

To train our algorithm, we use eight different genres: broadcast news (bn, 671 docs), broadcast conversations (bc, 698 docs), meetings (mt, 493 docs), newswire (nw, 471 docs), conversational telephone speech (sb, 890 docs), weblogs (wl, 543 docs), Amazon reviews of books, videogames and films (az train, 218 docs), and chat data (chat, 187 docs). To test our algorithm, we add six additional genres: Amazon reviews of appliances (az test, 27 docs), Wikipedia entries (wiki, 254 docs), Wikipedia discussion entries (wiki talk, 1792 docs), European Parliament transcripts (europarl, 1423 docs), a web collection obtained from Google searches for common conversational n-grams (web, 18540 docs), and transcribed McCain and Obama speeches (speeches, 20 docs). With the exception of the chat data, Amazon reviews, and a subset of the Europarl transcripts, the training corpora are from standard published datasets. The reviews, chat, Wikipedia, and web data were all collected from websites and cleaned locally. The documents average 600-1000 words in length, except for smaller corpora like Amazon reviews, whose documents average about 200 words. For training factored models, we assume that all the documents within a corpus share the same class.

2.2 Features and Classifier

The features used in (Feldman et al., 2009) were derived from a union of POS tags and a set of hand-picked, informative words. A similar approach is used here, including a collapsed version of the Treebank POS tag set (Marcus et al., 1993), with additions for specific words (e.g. personal pronouns and filled pause markers), compound punctuation (e.g. multiple exclamation marks), and a general emoticon tag, resulting in a total of 41 tags. Histograms are computed for a sliding window of length $w = 5$ over the tag sequence, and then statistics of each histogram bin are extracted. In the previous work, mean and standard deviation were extracted from the histogram bins. To this, we add skewness and kurtosis, which we will show are necessary for increased differentiation of unseen genres. For feature reduction, we used Principal Components Analysis and retained all PC dimensions with variance above 1% of the maximum PC variance.

Different approaches have been explored for computational modeling, including naive Bayes, linear discriminant modeling, and neural networks (Santini, 2004; Kessler et al., 1997; Stamatatos et al., 2000; Lee and Myaeng, 2002). Since (Feldman et al., 2009) found that quadratic discriminant analysis (QDA) outperforms naive Bayes, we use it here with full covariance matrices estimated by maximum likelihood, and trained on the reduced-dimension POS histogram features.

2.3 Factors

Linguistic research has tended to look at attributes of language rather than defining genre in terms of task domains. Since the number of task domains appears to be growing with new uses of the internet, we conjecture that an attribute approach is more practical for web-based text. We introduce the notion of a factored model for genre. The genre of each document can encoded as a vector of factors. Given data limits, the set of factors explored so far are:

- number of speakers/authors (1,2,3+),
- level of formality (low, medium, high),
- intended audience (personal, broadcast), and
- intent (inform, persuade).

Assuming factor independence, we train four separate QDA classifiers, one per factor. Using factors increases the richness of the space represented by the training set, in that it is possible to identify genres with factor combinations not seen in training.

3 Experiments and Discussion

3.1 Within-Domain Validation

As a preliminary step, and to ensure that the addition of skewness and kurtosis, as well as extra syntactic features, does not significantly impact the within-domain classification accuracy, we performed experiments with both the features in (Feldman et al., 2009) and our expanded features. For this, we split the training data 75/25 into training/test sets, and repeated the random split 50 times. We ran the experiments for both the original genre classification problem and the individual factors. We found that the addition of new moments and features decreased performance by less than 1% on average. We hypothesize that this small deterioration in performance is likely due to overtuning to the original training set.

3.2 New Features with Unseen Genres

To assess the use of our new features (added punctuation and emoticons) and the higher-order moments, we classified the web data with different processing configurations. In addition to the eight training genres, we introduced an "undetermined" genre or class for documents with a uniform posterior probability across all genres, which occurs when there is a large mismatch to all training genres. The distribution of labels is shown in Figure 1. While we do not have hand-labeled categories for this data, we thought it highly unlikely that the vast majority is bn, as predicted by the models using only mean and variance moments.

To validate our hypothesis that the spread of labels was more appropriate for the data, we randomly selected 100 documents and hand-labeled these using the eight classes plus "undetermined." The undetermined class was used for new genres (play scripts, lectures, newsgroups, congressional records). We found that it was difficult to annotate the data, since many samples had characteristics of more than one genre; this finding motivates the factor representation. The main difference between the various feature extraction configurations was in the detection of the undetermined cases. For the subset of undetermined documents that we labeled (34), none were detected using only 2 moments, but 35-40% were detected with the higher-order moments. Of the false detections, roughly 25-30% were associated with documents with characteristics of multiple classes. The effect of adding more detailed punctuation and emoticons to the tag set was not significant.

It should be noted that the web collection was based on queries designed to extract BC-style text, yet only 3 of 100 hand-labeled samples were in that category, none of which were accurately classified. Roughly 16 of the 100 documents are labeled as very informal and another 55 include some informal text or are moderately informal. This finding, combined with the observation that many documents reflect a mix of genres, suggests that a factored representation of genre (with formality as one "factor") may be more useful than explicit modeling of genres.

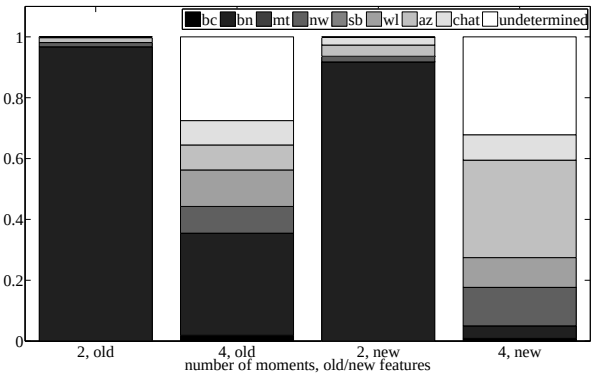


Figure 1: Fraction of web data classified as each genre.

3.3 Unseen Genre Factor Results

We trained a set of models for each factor and obtained posterior estimates for unseen classes. Figure 2 shows the class of out-of-domain documents for the formality factor, using 3 categories of formality: low (conversational, unprofessional), medium (casual but coherent), high (formal). We have not hand-labeled individual documents in all of these sets, but the resulting class proportions match our intuition for these genres. The Wikipedia data is labeled as highly formal, and most web data is labeled as medium. Examining the 100 hand-labeled web documents, we find that adding the higher-order moments improves classifier accuracy from 23% to 55%. The effect of the added tag set features was once again not significant.

Figure 3 shows the class of out-of-domain documents for the factor indicating number of speakers/authors. This factor appears difficult to detect.

We hypothesize that there is an unaccounted-for dependence on audience. When there is a listener, speakers may use the term "you," as in conversations and internet chat. An interesting observation is that the ten Obama speeches all appear to exhibit this behavior. McCain speeches, on the other hand, display some variation, and about a third are (correctly) characterized as single speaker.

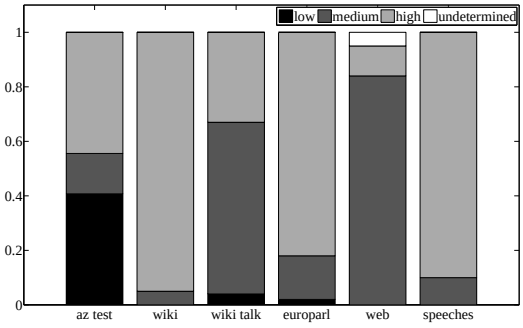


Figure 2: Test corpora classification, formality.

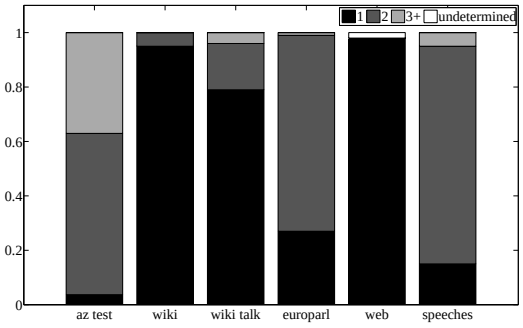


Figure 3: Test corpora classification, number of speakers.

The audience factor results are very skewed towards broadcast data, but this matches our intuition, and the scarcity of data meant for private consumption, so they are not included. However, further study is needed, since 3-dimensional projections of the training data suggest a Gaussian mixture (or other more complex model) may fit better.

The intent factor results are also mixed. The classifier labels most of the Wikipedia, europarl, web, and speeches data as "report," and most reviews as "persuade." While the "report" category fits Wikipedia, it is not clear that europarl should also be classified as "report," since parliamentary proceedings are notoriously argumentative. With this factor, the noise inherent in using genre-level labels is significant. It is not always clear what is reportage and what is persuasion, and we expect some genres (e.g. reviews) to be a mixture of both.

4 Summary

We have introduced new features that are more robust for handling domains unseen in training, and presented a factored genre framework that allows for a finer-grained representation of genre. Many open questions remain, including which other factors can or cannot be captured by our current feature set and classifier, and whether noisy label learning methods could address the problem of uncertainty in the labels for particular features and genres.

References

D. Biber. 1993. Using register-diversified corpora for general language studies. *Computational Linguistics*, 19(2):219–242.

I. Bulyko et al. 2007. Web resources for language modeling in conversational speech recognition. *ACM Trans. on Speech and Language Processing*, 5(1):1–25.

S. Feldman et al. 2009. Part-of-speech histograms for genre classification of text. *Proc. ICASSP*.

W. Wang. 2008. Weakly supervised training for parsing mandarin broadcast transcripts. *Proc. Interspeech.*

J. Karlgren and D. Cutting. 1994. Recognizing text genres with simple metrics using discriminant analysis. *Proc. Computational Linguistics*, pages 1071–1075.

B. Kessler, G. Numberg, and H. Schütze. 1997. Automatic detection of text genre. *ACL-35*, pages 32–38.

Y.-B. Lee and S. H. Myaeng. 2002. Text genre classification with genre-revealing and subject-revealing features. *ACM SIGIR*, pages 145–150.

C. S. Lim, K. J. Lee, and G. C. Kim. 2004. Automatic genre detection of web documents. *IJCNLP*.

M. P. Marcus et al. 1993. Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

M. Santini. 2004. A shallow approach to syntactic feature extraction for genre classification. *CLUK 7: UK special-interest group for computational linguistics*.

E. Stamatatos, N. Fakotakis, and G. Kokkinakis. 2000. Text genre detection using common word frequencies. *Proc. Computational Linguistics*, pages 808–814.

Towards Effective Sentence Simplification for Automatic Processing of Biomedical Text

Siddhartha Jonnalagadda[*], **Luis Tari**[**], **Jörg Hakenberg**[**], **Chitta Baral**[**], **Graciela Gonzalez**[*]
[*]Department of Biomedical Informatics, Arizona State University, Phoenix, AZ 85004, USA.
[**]Department of Computer Science and Engineering, Arizona State University, Tempe, AZ 85281, USA.
Corresponding author: ggonzalez@asu.edu

Abstract

The complexity of sentences characteristic to biomedical articles poses a challenge to natural language parsers, which are typically trained on large-scale corpora of non-technical text. We propose a text simplification process, bioSimplify, that seeks to reduce the complexity of sentences in biomedical abstracts in order to improve the performance of syntactic parsers on the processed sentences. Syntactic parsing is typically one of the first steps in a text mining pipeline. Thus, any improvement in performance would have a ripple effect over all processing steps. We evaluated our method using a corpus of biomedical sentences annotated with syntactic links. Our empirical results show an improvement of 2.90% for the Charniak-McClosky parser and of 4.23% for the Link Grammar parser when processing simplified sentences rather than the original sentences in the corpus.

1 Introduction

It is typical that applications for biomedical text involve the use of natural language syntactic parsers as one of the first steps in processing. Thus, the performance of the system as a whole is largely dependent on how well the natural language syntactic parsers perform.

One of the challenges in parsing biomedical text is that it is significantly more complex than articles in typical English text. Different analysis show other problematic characteristics, including inconsistent use of nouns and partial words (Tateisi & Tsujii, 2004), higher perplexity measures (Elhadad, 2006), greater lexical density, plus increased number of relative clauses and prepositional phrases (Gemoets, 2004), all of which correlate with diminished comprehension and higher text difficulty. These characteristics also lead to performance problems in terms of computation time and accuracy for parsers that are trained on common English text corpus.

We identified three categories of sentences: 1) normal English sentences, like in Newswire text, 2) normal biomedical English sentences – those sentences which can be parsed without a problem by Link Grammar-, and 3) complex biomedical English sentences – those sentences which can't be parsed by Link Grammar. Aside from the known characteristics mentioned before, sentences in the third group tended to be longer (18% of them had more than 50 words, while only 8% of those in group 2 and 2% of those in group 1 did). It has been observed that parsers perform well with sentences of reduced length (Chandrasekar & Srinivas, 1997; Siddharthan, 2006).

In this paper, we explore the use of text simplification as a preprocessing step for general parsing to reduce length and complexity of biomedical sentences in order to enhance the performance of the parsers.

2 Methods

There are currently many publicly available corpora of biomedical texts, the most popular among them being BioInfer, Genia, AImed, HPRD 50, IEPA, LLL and BioCreative1-PPI. Among these corpora, BioInfer includes the most comprehensive collection of sentences and careful annotation for links of natural parser, in both the Stanford and Link Grammar schemes. Therefore, we chose the BioInfer corpus, version 1.1.0 (Pyysalo et al., 2007), containing 1100 sentences for evaluating the effectiveness of our simplification method on

Proceedings of NAACL HLT 2009: Short Papers, pages 177–180,
Boulder, Colorado, June 2009.

the performance of syntactic parsers. The method includes syntactic and non-syntactic transformations, detailed next.

2.1 Non-syntactic transformation

We group here three steps of our approach: 1. preprocessing through removal of spurious phrases; 2. replacement of gene names; 3. replacement of noun phrases.

To improve the correctness of the parsing, each biomedical sentence is first preprocessed to remove phrases that are not essential to the sentence. This includes removal of *section indicators*, which are phrases that specify the name of the section at the beginning of the sentence, plus the removal of phrases in parentheses (such as citations and numbering in lists). Also, partially hyphenated words are transformed by combining with the nearest word that follows or precedes the partial hyphenated word to make a meaningful word. For instance, the phrase "alpha- and beta-catenin" is transformed into "alpha-catenin and beta-catenin".

Occurrences of multi-word technical terms and entity names involved in biomedical processes are common in biomedical text. Such terms are not likely to appear in the dictionary of a parser (perplexity is high), and will force it to use morpho-guessing and unknown word guessing. This is time consuming and prone to error. Thus, unlike typical text simplification that emphasizes syntactic transformation of sentences, our approach utilizes a named entity recognition engine, BANNER (Leaman & Gonzalez, 2008), to replace multi-word gene names with single-word placeholders.

Replacement of gene names with single elements is not enough, however, and grammatical category (i.e. singular or plural) of the element has to be considered. Lingpipe (Alias-i, 2006), a shallow parser for biomedical text, identifies noun phrases and replaces them with single elements. A single element is considered singular when the following verb indicates a third-person singular verb or the determiner preceded by the element is either "a" or "an". Otherwise it is considered as plural and an "s" is attached to the end of the element.

2.2 Syntactic transformation

The problem of simplifying long sentences in common English text has been studied before, notably by Chandrasekar & Srinivas (1997) and Siddharthan (2006). However, the techniques used in these studies might not totally solve the issue of parsing biomedical sentences. For example, using Siddharthan's approach, the biological finding "The Huntington's disease protein interacts with p53 and CREB-binding protein and represses transcription", and assuming multi-word nouns such as "CREB-binding protein" do not present a problem, would be simplified to:

> *"The Huntington's disease protein interacts with p53. The Huntington's disease protein interacts with CREB-binding protein. The Huntington's disease protein represses transcription."*

Our method transforms it to "*GENE1 interacts with GENE2 and GENE3 and represses transcription.*" Both decrease the average sentence length, but the earlier distorts the biological meaning (since the Huntington's disease protein might not repress transcription on its own), while the latter signifies it.

While replacement of gene names and noun phrases can reduce the sentence length, there are

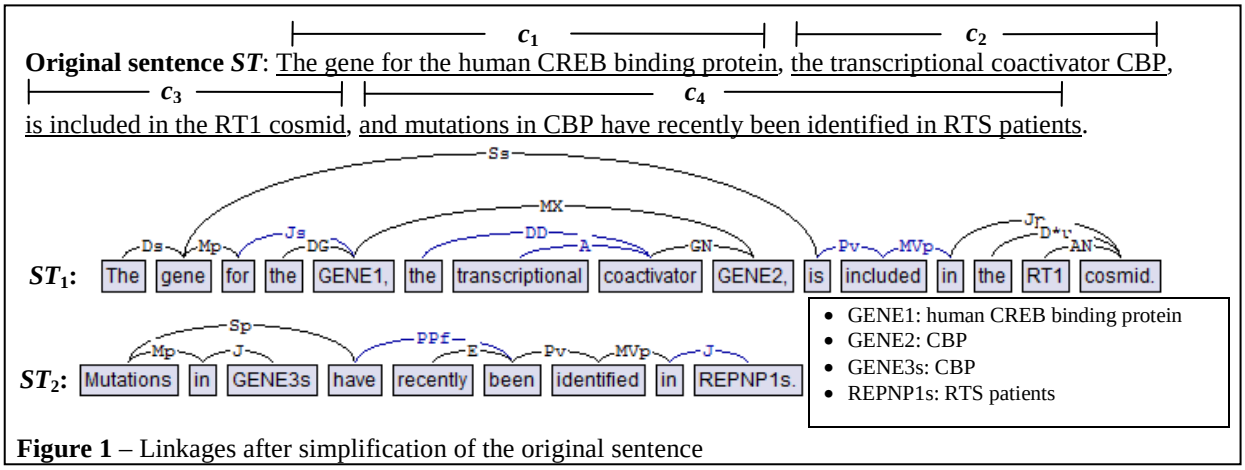


Figure 1 – Linkages after simplification of the original sentence

cases when the sentences are still too complex to be parsed efficiently. We developed a simple algorithm that utilizes *linkages* (specific grammatical relationships between pairs of words in a sentence) of the Link Grammar parser (Sleator, 1998) and punctuations for splitting sentences into clauses. An example in Figure 1 illustrates the main part of the algorithm. Each linkage has a primary *link type* in CAPITAL followed by secondary link type in short. The intuition behind the algorithm is to try to identify independent clauses from complex sentences. The first step is to split the sentence *ST* into clauses c_1, c_2, c_3 and c_4 based on commas. c_1 is parsed using the Link Grammar parser, but c_1 cannot be a sentence as there is no "S" link in the linkage of c_1. c_2 is then attached to c_1 and the linkage of "c_1, c_2" does not contain a "S" link as well. "c_1, c_2, c_3." is recognized as a sentence, since the linkage contains an "S" link, indicating that it is a sentence, as well as the linkage of c_4. So the algorithm returns two sentences ST_1 and ST_2 for *ST*.

3 Results

Our method has the greatest impact on the performance of Link Grammar (LG), which lies at the core of BioLG (Pyysalo et al., 2006). However, it also has a significant impact on the self-training biomedical parser by McClosky & Charniak (CM), which is currently the best parser available for biomedical text.

3.1 Rudimentary statistics of the results of simplification: After the simplification algorithm was tested on the 1100 annotated sentences of the BioInfer corpus, there were 1159 simplified sentences because of syntactic transformation (section 2.2). The number of words per sentence showed a sharp drop of 20.4% from 27.0 to 21.5. The Flesh-Kincaid score for readability dropped from 17.4 to 14.2. The Gunning Fog index also dropped by 18.3% from 19.7 to 16.1.

Pre-processing	Replacement of gene names	Replacement of noun phrases	Syntactic Simplification
359	1082	915	91

Table 1: Sentences processed in each stage

3.2 Impact of simplification on the efficiency of parsing: We inputted the BioInfer corpus to LG and CM. If LG cannot find a complete linkage, it invokes its *panic mode*, where sentences are returned with considerably low accuracy. Out of the 1100 original sentences in the corpus, 219 went into panic mode. After processing, only 39 out of 1159 simplified sentences triggered panic mode (a 16.4% improvement in efficiency). The average time for parsing a sentence also dropped from 7.36 secs to 1.70 secs after simplification.

3.3 Impact of simplification on the accuracy of parsing: Let Σg, Σo and Σs, respectively be the sets containing the links of the gold standard, the output generated by the parser on original sentences and the output generated by the parser on simplified sentences. We denote a link of type Π between the tokens Φ_1 and Φ_2 by (Π,Φ_1,Φ_2). In the case of the original sentences, the tokens $\Phi_1$and Φ_2 are single-worded. So, (Π,Φ_1,Φ_2) is a true positive iff (Π,Φ_1,Φ_2) belongs to both Σg and Σo, false positive iff it only belongs to Σo and false negative iff it only belongs to Σg. In the case of simplified sentences, the tokens $\Phi_1$and Φ_2 can have multiple words. So, (Π,Φ_1,Φ_2) which belongs to Σs is a true positive iff (Π,Φ'_1,Φ'_2) belongs to Σg where Φ'_1 and Φ'_2 are respectively one of the words in Φ_1 and Φ_2. Additionally, (Π,Φ_1,Φ_2) which belongs to Σg is not a false negative if Φ_1 and Φ_2 are parts of a single token of a simplified sentence. For measuring the performance of a parser, the nature of linkage is most relevant in the context of the sentence in consideration. So, we calculate precision and recall for each sentence and average them over all sentences to get the respective precision and recall for the collection.

	Precision	**Recall**	**f-measure**
CM	77.94%	74.08%	75.96%
BioSimplify + CM	82.51%	75.51%	78.86%
Improvement	**4.57%**	**1.43%**	**2.90%**
LG	72.36%	71.65%	72.00%
BioSimplify + LG	78.30%	74.27%	76.23%
Improvement	**5.94%**	**2.62%**	**4.23%**

Table 2: Accuracy of McClosky & Charniak (CM) and Link Grammar (LG) parsers based on Stanford dependencies, with and without simplified sentences.

In order to compare the effect of BioSimplify on the two parsers, a converter from Link Grammar to Stanford scheme was used (Pyysalo et al, 2007: precision and recall of 98% and 96%). Results of

this comparision are shown in Table 2. On CM and LG, we were able to achieve a considerable improvement in the f-measures by 2.90% and 4.23% respectively. CM demonstrated an absolute error reduction of 4.1% over its previous best on a different test set. Overall, bioSimplify leverages parsing of biomedical sentences, increasing both the efficiency and accuracy.

4 Related work

During the creation of BioInfer, noun phrase macro-dependencies were determined using a simple rule set without parsing. Some of the problems related to parsing noun phrases were removed by reducing the number of words by more than 20%. BioLG enhances LG by expansion of lexicons and the addition of morphological rules for biomedical domain. Our work differs from BioLG not only in utilizing a gene name recognizer, a specialized shallow parser and syntactic transformation, but also in creating a preprocessor that can improve the performance of any parser on biomedical text.

The idea of improving the performance of deep parsers through the integration of shallow and deep parsers has been reported in (Crysmann et al., 2002; Daum et al., 2003; Frank et al., 2003) for non-biomedical text. In BioNLP, extraction systems (Jang et al., 2006; Yakushiji et al., 2001) used shallow parsers to enhance the performance of deep parsers. However, there is a lack of evaluation of the correctness of the dependency parses, which is crucial to the correctness of the extracted systems. We not only evaluate the correctness of the links, but also go beyond the problem of relationship extraction and empower future researchers in leveraging their parsers (and other extraction systems) to get better results.

5 Conclusion and Future work

We achieved an f-measure of 78.86% using CM on BioInfer Corpus which is a 2.90% absolute reduction in error. We achieved a 4.23% absolute reduction in error using LG. According to the measures described in section 3.1, the simplified sentences of BioInfer outperform the original ones by more than 18%. Our method can also be used with other parsers. As future work, we will demonstrate the impact of our simplification method on other text mining tasks, such as relationship extraction.

Acknowledgments

We thank Science Foundation Arizona (award CAA 0277-08 Gonzalez) for partly supporting this research. SJ also thanks Bob Leaman and Anoop Grewal for their guidance.

References

Chandrasekar, R., & Srinivas, B. (1997). Automatic induction of rules for text simplification. *Knowledge-Based Systems, 10*, 183-190.

Crysmann, B., Frank, A., Kiefer, B., Muller, S., Neumann, G., et al. (2002). An integrated architecture for shallow and deep processing. *ACL'02*.

Daum, M., Foth, K., & Menzel, W. (2003). Constraint based integration of deep and shallow parsing techniques. *EACL'03*.

Elhadad, Noémie (2006) User-Sensitive Text Summarization: Application to the Medical Domain. Ph.D. Thesis, Columbia University. Available at www.dbmi.columbia.edu/noemie/papers/thesis.pdf

Frank, A., Becker M, et al.,. (2003). Integrated shallow and deep parsing: TopP meets HPSG. *ACL'03*.

Gemoets, D., Rosemblat, G., Tse, T., Logan, R., Assessing Readability of Consumer Health Information: An Exploratory Study. MEDINFO 2004.

Jang, H., Lim, J., Lim, J.-H., Park, S.-J., Lee, K.-C. and Park, S.-H. (2006) Finding the evidence for protein-protein interactions from PubMed abstracts, *Bioinformatics*, **22**, e220-226.

Leaman, R., & Gonzalez, G. (2008). BANNER: An executable survery of advances in biomedical named entity recognition. 652-663.

Manning, C. D., & Schütze, H. (1999). *Foundations of statistical natural language processing* MIT Press.

McClosky, D., & Charniak, E. (2008) Self-training for biomedical parsing. *ACL'08*.

Pyysalo, S., Ginter, F., Haverinen, K., Heimonen, J., Salakoski, T., & Laippala, V. (2007) On the unification of syntactic annotations under the stanford dependency scheme.. *ACL'07*.

Pyysalo, S., Salakoski, T., Aubin, S., & Nazarenko, A. (2006). Lexical adaptation of link grammar to the biomedical sublanguage: A comparative evaluation of three approaches. *BMC Bioinformatics, 7*, S2.

Siddharthan, A. (2006). Syntactic simplification and text cohesion. *Res Lang Comput, 4*(1), 77-109.

Sleator, D. (1998) Link Grammar Documentation

Tateisi, Y., & Tsujii, J. (2004). Part-of-speech annotation of biology research abstracts. *LREC*, 1267-1270.

Yakushiji, A., Tateisi, Y., Miyao, Y., & Tsujii, J. (2001). Event extraction from biomedical papers using a full parser. *PSB'01, 6*, 408-419.

Improving SCL Model for Sentiment-Transfer Learning

Songbo Tan
Institute of Computing Technology
Beijing, China
tansongbo@software.ict.ac.cn

Xueqi Cheng
Institute of Computing Technology
Beijing, China
cxq@ict.ac.cn

ABSTRACT

In recent years, Structural Correspondence Learning (SCL) is becoming one of the most promising techniques for sentiment-transfer learning. However, SCL model treats each feature as well as each instance by an equivalent-weight strategy. To address the two issues effectively, we proposed a weighted SCL model (W-SCL), which weights the features as well as the instances. More specifically, W-SCL assigns a smaller weight to high-frequency domain-specific (HFDS) features and assigns a larger weight to instances with the same label as the involved pivot feature. The experimental results indicate that proposed W-SCL model could overcome the adverse influence of HFDS features, and leverage knowledge from labels of instances and pivot features.

1 Introduction

In the community of sentiment analysis (Turney 2002; Pang et al., 2002; Tang et al., 2009), transferring a sentiment classifier from one source domain to another target domain is still far from a trivial work, because sentiment expression often behaves with strong domain-specific nature.

Up to this time, many researchers have proposed techniques to address this problem, such as classifiers adaptation, generalizable features detection and so on (DaumeIII et al., 2006; Jiang et al., 2007; Tan et al., 2007; Tan et al., 2008; Tan et al., 2009). Among these techniques, SCL (Structural Correspondence Learning) (Blitzer et al., 2006) is regarded as a promising method to tackle transfer-learning problem. The main idea behind SCL model is to identify correspondences among features from different domains by modeling their correlations with pivot features (or generalizable features). Pivot features behave similarly in both domains. If non-pivot features from different domains are correlated with many of the same pivot features, then we assume them to be corresponded with each other, and treat them similarly when training a sentiment classifier.

However, SCL model treats each feature as well as each instance by an equivalent-weight strategy. From the perspective of feature, this strategy fails to overcome the adverse influence of high-frequency domain-specific (HFDS) features. For example, the words "stock" or "market" occurs frequently in most of stock reviews, so these non-sentiment features tend to have a strong correspondence with pivot features. As a result, the representative ability of the other sentiment features will inevitably be weakened to some degree.

To address this issue, we proposed Frequently Exclusively-occurring Entropy (FEE) to pick out HFDS features, and proposed a feature-weighted SCL model (FW-SCL) to adjust the influence of HFDS features in building correspondence. The main idea of FW-SCL is to assign a smaller weight to HFDS features so that the adverse influence of HFDS features can be decreased.

From the other perspective, the equivalent-weight strategy of SCL model ignores the labels ("positive" or "negative") of labeled instances. Obviously, this is not a good idea. In fact, positive pivot features tend to occur in positive instances, so the correlations built on positive instances are more reliable than that built on negative instances; and vice versa. Consequently, utilization of labels of instances and pivot features can decrease the adverse influence of some co-occurrences, such as co-occurrences involved with positive pivot features and negative instances, or involved with negative pivot features and positive instances.

In order to take into account the labels of labeled instances, we proposed an instance-weighted SCL model (IW-SCL), which assigns a larger weight to instances with the same label as the involved pivot feature. In this time, we obtain a combined model: feature-weighted and instance-weighted SCL model (FWIW-SCL). For the sake

Proceedings of NAACL HLT 2009: Short Papers, pages 181–184,
Boulder, Colorado, June 2009.

of convenience, we simplify "FWIW-SCL" as "W-SCL" in the rest of this paper.

2 Structural Correspondence Learning

In the section, we provide the detailed procedures for SCL model.

First we need to pick out pivot features. Pivot features occur frequently in both the source and the target domain. In the community of sentiment analysis, generalizable sentiment words are good candidates for pivot features, such as "good" and "excellent". In the rest of this paper, we use K to stand for the number of pivot features.

Second, we need to compute the pivot predictors (or mapping vectors) using selected pivot features. The pivot predictors are the key job, because they directly decide the performance of SCL. For each pivot feature k, we use a loss function L_k,

$$L_k = \sum_i \left(p_k(x_i) w^T x_i - 1\right) + \lambda \|w\|^2 \qquad (1)$$

where the function $p_k(x_i)$ indicates whether the pivot feature k occurs in the instance x_i,

$$p_k(x_i) = \begin{cases} 1 & if\ x_{ik} > 0 \\ -1 & otherwise \end{cases},$$

where the weight vector w encodes the correspondence of the non-pivot features with the pivot feature k (Blitzer et al., 2006).

Finally we use the augmented space $[x^T, x^T W]^T$ to train the classifier on the source labeled data and predict the examples on the target domain, where $W=[w_1, w_2, \ldots, w_K]$.

3 Feature-Weighted SCL Model

3.1 Measure to pick out HFDS features

In order to pick out HFDS features, we proposed Frequently Exclusively-occurring Entropy (FEE). Our measure includes two criteria: occur in one domain as frequently as possible, while occur on another domain as rarely as possible. To satisfy this requirement, we proposed the following formula:

$$f_w = \log\left(\max\left(P_o(w), P_n(w)\right)\right) + \log\left(\frac{\max\left(P_o(w), P_n(w)\right)}{\min\left(P_o(w), P_n(w)\right)}\right) \qquad (2)$$

where $P_o(w)$ and $P_n(w)$ indicate the probability of word w in the source domain and the target domain respectively:

$$P_o(w) = \frac{\left(N_o(w) + \alpha\right)}{\left(N_o + 2 \cdot \alpha\right)} \qquad (3)$$

$$P_n(w) = \frac{\left(N_n(w) + \alpha\right)}{\left(N_n + 2 \cdot \alpha\right)} \qquad (4)$$

where $N_o(w)$ and $N_n(w)$ is the number of examples with word w in the source domain and the target domain respectively; N_o and N_n is the number of examples in the source domain and the target domain respectively. In order to overcome overflow, we set α=0.0001 in our experiment reported in section 5.

To better understand this measure, let's take a simple example (see Table 1). Given a source dataset with 1000 documents and a target dataset with 1000 documents, 12 candidate features, and a task to pick out 2 HFDS features. According to our understanding, the best choice is to pick out w_4 and w_8. According to formula (2), fortunately, we successfully pick out w_4, and w_8. This simple example validates the effectiveness of proposed FEE formula.

Table 1: A simple example for FEE

Words	$N_o(w)$	$N_n(w)$	*FEE*	
			Score	*Rank*
w_1	100	100	-2.3025	6
w_2	100	90	-2.1971	4
w_3	100	45	-1.5040	3
w_4	**100**	**4**	**0.9163**	**1**
w_5	50	50	-2.9956	8
w_6	50	45	-2.8903	7
w_7	50	23	-2.2192	5
w_8	**50**	**2**	**0.2231**	**2**
w_9	4	4	-5.5214	11
w_{10}	4	3	-5.2337	10
w_{11}	4	2	-4.8283	9
w_{12}	1	1	-6.9077	12

3.2 Feature-Weighted SCL model

To adjust the influence of HFDS features in building correspondence, we proposed feature-weighted SCL model (FW-SCL),

$$L_k = \sum_i \left(p_k(x_i) \sum_l \delta_l w_l x_{il} - 1\right) + \lambda \|w\|^2 \qquad (5)$$

where the function $p_k(x_i)$ indicates whether the pivot feature k occurs in the instance x_i;

$$p_k(x_i) = \begin{cases} 1 & if\ x_{ik} > 0 \\ -1 & otherwise \end{cases},$$

and δ_l is the parameter to control the weight of the HFDS feature l,

$$\delta_l = \begin{cases} \eta & if \ \ l \in Z_{HFDS} \\ 1 & otherwise \end{cases}$$

where Z_{HFDS} indicates the HFDS feature set and η is located in the range $[0,1]$. When "η=0", it indicates that no HFDS features are used to build the correspondence vectors; while "η=1" indicates that all features are equally used to build the correspondence vectors, that is to say, proposed FW-SCL algorithm is simplified as traditional SCL algorithm. Consequently, proposed FW-SCL algorithm could be regarded as a generalized version of traditional SCL algorithm.

4 Instance-Weighted SCL Model

The traditional SCL model does not take into account the labels ("positive" or "negative") of instances on the source domain and pivot features. Although the labels of pivot features are not given at first, it is very easy to obtain these labels because the number of pivot features is typically very small.

Obviously, positive pivot features tend to occur in positive instances, so the correlations built on positive instances are more reliable than the correlations built on negative instances; and vice versa. As a result, the ideal choice is to assign a larger weight to the instances with the same label as the involved pivot feature, while assign a smaller weight to the instances with the different label as the involved pivot feature. This strategy can make correlations more reliable. This is the key idea of instance-weighted SCL model (IW-SCL). Combining the idea of feature-weighted SCL model (FW-SCL), we obtain the feature-weighted and instance-weighted SCL model (FWIW-SCL),

$$L_k = \gamma \cdot \sum \rho(\psi(k),\psi(x_i))(p_k(x_i)\sum_l \delta_l w_l x_{il} - 1) + \lambda\|w\|^2 + (1-\gamma)\cdot \sum (1-\rho(\psi(k),\psi(x_j)))(p_k(x_j)\sum_l \delta_l w_l x_{jl} - 1) \quad (6)$$

where γ is the instance weight and the function $p_k(x_i)$ indicates whether the pivot feature k occurs in the instance x_i;

$$p_k(x_i) = \begin{cases} 1 & if \ x_{ik} > 0 \\ -1 & otherwise \end{cases}$$

and δ_l is the parameter to control the weight of the HFDS feature l,

$$\delta_l = \begin{cases} \eta & if \ \ l \in Z_{HFDS} \\ 1 & otherwise \end{cases},$$

where Z_{HFDS} indicates the HFDS feature set and η is located in the range $[0,1]$.

In equation (6), the function $\rho(z,y)$ indicates whether the two variables z and y have the same non-zero value,

$$\rho(z,y) = \begin{cases} 1 & if \ \ z = y \ and \ z \neq 0 \\ 0 & otherwise \end{cases};$$

and the function $\psi(z)$ is a hinge function, whose variables are either pivot features or instances,

$$\psi(z) = \begin{cases} 1 & if \ \ z \ has \ a \ \ positive \ \ label \\ 0 & unknown \\ -1 & if \ \ z \ has \ a \ \ negative \ \ label \end{cases}.$$

For the sake of convenience, we simplify "FWIW-SCL" as "W-SCL".

5 Experimental Results

5.1 Datasets

We collected three Chinese domain-specific datasets: Education Reviews (Edu, from http://blog.sohu.com/learning/), Stock Reviews (Sto, from http://blog.sohu.com/stock/) and Computer Reviews (Comp, from http://detail.zol.com.cn/). All of these datasets are annotated by three linguists. We use ICTCLAS (a Chinese text POS tool, http://ictclas.org/) to parse Chinese words.

The dataset Edu includes 1,012 negative reviews and 254 positive reviews. The average size of reviews is about 600 words. **The dataset Sto** consists of 683 negative reviews and 364 positive reviews. The average length of reviews is about 460 terms. **The dataset Comp** contains 390 negative reviews and 544 positive reviews. The average length of reviews is about 120 words.

5.2 Comparison Methods

In our experiments, we run one supervised baseline, i.e., Naïve Bayes (NB), which only uses one source-domain labeled data as training data.

For transfer-learning baseline, we implement traditional SCL model (T-SCL) (Blitzer et al., 2006). Like TSVM, it makes use of the source-domain labeled data as well as the target-domain unlabeled data.

5.3 Does proposed method work?

To conduct our experiments, we use source-domain data as unlabeled set or labeled training set, and use target-domain data as unlabeled set or testing set. Note that we use 100 manual-annotated pivot features for T-SCL, FW-SCL and W-SCL in the following experiments. We select

pivot features use three criteria: a) is a sentiment word; b) occurs frequently in both domains; c) has similar occurring probability. For T-SCL, FW-SCL and W-SCL, we use prototype classifier (Sebastiani, 2002) to train the final model.

Table 2 shows the results of experiments comparing proposed method with supervised learning, transductive learning and T-SCL. For FW-SCL, the Z_{HFDS} is set to 200 and η is set to 0.1; For W-SCL, the Z_{HFDS} is set to 200, η is set to 0.1, and γ is set to 0.9.

As expected, proposed method FW-SCL does indeed provide much better performance than supervised baselines, TSVM and T-SCL model. For example, the average accuracy of FW-SCL beats supervised baselines by about 12 percents, beats TSVM by about 11 percents and beats T-SCL by about 10 percents. This result indicates that proposed FW-SCL model could overcome the shortcomings of HFDS features in building correspondence vectors.

More surprisingly, instance-weighting strategy can further boost the performance of FW-SCL by about 4 percents. This result indicates that the labels of instances and pivot features are very useful in building the correlation vectors. This result also verifies our analysis in section 4: positive pivot features tend to occur in positive instances, so the correlations built on positive instances are more reliable than the correlations built on negative instances, and vice versa.

Table 2: Accuracy of different methods

	NB	T-SCL	FW-SCL	W-SCL
Edu->Sto	0.6704	0.7965	0.7917	0.8108
Edu->Comp	0.5085	0.8019	0.8993	0.9025
Sto->Edu	0.6824	0.7712	0.9072	0.9368
Sto->Comp	0.5053	0.8126	0.8126	0.8693
Comp->Sto	0.6580	0.6523	0.7010	0.7717
Comp->Edu	0.6114	0.5976	0.9112	0.9408
Average	0.6060	0.7387	0.8372	0.8720

Although SCL is a method designed for transfer learning, but it cannot provide better performance than TSVM. This result verifies the analysis in section 3: a small amount of HFDS features occupy a large amount of *weight* in classification model, but hardly carry corresponding sentiment. In another word, very few top-frequency words degrade the representative ability of SCL model for sentiment classification.

6 Conclusion Remarks

In this paper, we proposed a weighted SCL model (W-SCL) for domain adaptation in the context of sentiment analysis. On six domain-transfer tasks, W-SCL consistently produces much better performance than the supervised, semi-supervised and transfer-learning baselines. As a result, we can say that proposed W-SCL model offers a better choice for sentiment-analysis applications that require high-precision classification but hardly have any labeled training data.

7 Acknowledgments

This work was mainly supported by two funds, i.e., 0704021000 and 60803085, and one another project, i.e., 2004CB318109.

References

Blitzer, J. and McDonald, R. and Fernando Pereira. Domain adaptation with structural correspondence learning. EMNLP 2006.

DaumeIII, H. and Marcu, D. Domain adaptation for statistical classifiers. Journal of Artificial Intelligence Research, 2006, 26: 101-126.

Jiang, J., Zhai, C. A Two-Stage Approach to Domain Adaptation for Statistical Classifiers. CIKM 2007.

Pang, B., Lee, L. and Vaithyanathan, S. Thumbs up? Sentiment classification using machine learning techniques. EMNLP 2002.

Sebastiani, F. Machine learning in automated text categorization. ACM Computing Surveys. 2002, 34(1): 1-47.

S. Tan, G. Wu, H. Tang and X. Cheng. A novel scheme for domain-transfer problem in the context of sentiment analysis. CIKM 2007.

S. Tan, Y. Wang, G. Wu and X. Cheng. Using unlabeled data to handle domain-transfer problem of semantic detection. SAC 2008.

S. Tan, X. Cheng, Y. Wang and H. Xu. Adapting Naive Bayes to Domain Adaptation for Sentiment Analysis. ECIR 2009.

H. Tang, S. Tan, and X. Cheng. A Survey on Sentiment Detection of Reviews. Expert Systems with Applications. Elsevier. 2009, doi:10.1016/j.eswa.2009.02.063.

Turney, P. D. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. ACL 2002.

MICA: A Probabilistic Dependency Parser Based on Tree Insertion Grammars

Application Note

Srinivas Bangalore
AT&T Labs – Research
Florham Park, NJ, USA
srini@research.att.com

Pierre Boulllier
INRIA
Rocquencourt, France
Pierre.Boullier@inria.fr

Alexis Nasr
Aix-Marseille Université
Marseille, France
alexis.nasr@lif.univ-mrs.fr

Owen Rambow
CCLS, Columbia Univserity
New York, NY, USA
rambow@ccls.columbia.edu

Benoît Sagot
INRIA
Rocquencourt, France
benoit.sagot@inria.fr

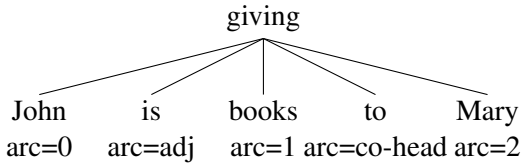
Abstract

MICA is a dependency parser which returns deep dependency representations, is fast, has state-of-the-art performance, and is freely available.

1 Overview

This application note presents a freely available parser, MICA (Marseille-INRIA-Columbia-AT&T).[1] MICA has several key characteristics that make it appealing to researchers in NLP who need an off-the-shelf parser.

• MICA returns a deep dependency parse, in which dependency is defined in terms of lexical predicate-argument structure, not in terms of surface-syntactic features such as subject-verb agreement. Function words such as auxiliaries and determiners depend on their lexical head, and strongly governed prepositions (such as *to* for *give*) are treated as co-heads rather than as syntactic heads in their own right. For example, *John is giving books to Mary* gets the following analysis (the arc label is on the terminal).



The arc labels for the three arguments *John*, *books*, and *Mary* do not change when the sentence is passivized or *Mary* undergoes dative shift.

[1]We would like to thank Ryan Roth for contributing the MALT data.

• MICA is based on an explicit phrase-structure tree grammar extracted from the Penn Treebank. Therefore, MICA can associate dependency parses with rich linguistic information such as voice, the presence of empty subjects (PRO), *wh*-movement, and whether a verb heads a relative clause.

• MICA is fast (450 words per second plus 6 seconds initialization on a standard high-end machine on sentences with fewer than 200 words) and has state-of-the-art performance (87.6% unlabeled dependency accuracy, see Section 5).

• MICA consists of two processes: the supertagger, which associates tags representing rich syntactic information with the input word sequence, and the actual parser, which derives the syntactic structure from the n-best chosen supertags. Only the supertagger uses lexical information, the parser only sees the supertag hypotheses.

• MICA returns n-best parses for arbitrary n; parse trees are associated with probabilities. A packed forest can also be returned.

• MICA is freely available[2], easy to install under Linux, and easy to use. (Input is one sentence per line with no special tokenization required.)

There is an enormous amount of related work, and we can mention only the most salient, given space constraints. Our parser is very similar to the work of (Shen and Joshi, 2005). They do not employ a supertagging step, and we do not restrict our trees to spinal projections. Other parsers using supertagging include the LDA of Bangalore and Joshi (1999), the CCG-based parser of Clark and Curran (2004), and the constraint-based approach of Wang

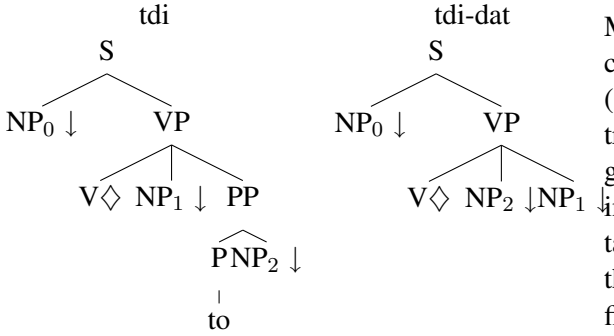
[2]http://www1.ccls.columbia.edu/~rambow/mica.html

Proceedings of NAACL HLT 2009: Short Papers, pages 185–188,
Boulder, Colorado, June 2009.

and Harper (2004). Widely used dependency parsers which generate deep dependency representations include Minipar (Lin, 1994), which uses a declarative grammar, and the Stanford parser (Levy and Manning, 2004), which performs a conversion from a standard phrase-structure parse. All of these systems generate dependency structures which are slightly different from MICA's, so that direct comparison is difficult. For comparison purposes, we therefore use the MALT parser generator (Nivre et al., 2004), which allows us to train a dependency parser on our own dependency structures. MALT has been among the top performers in the CoNLL dependency parsing competitions.

2 Supertags and Supertagging

Supertags are elementary trees of a lexicalized tree grammar such as a Tree-Adjoining Grammar (TAG) (Joshi, 1987). Unlike context-free grammar rules which are single level trees, supertags are multi-level trees which encapsulate both predicate-argument structure of the anchor lexeme (by including nodes at which its arguments must substitute) and morpho-syntactic constraints such as subject-verb agreement within the supertag associated with the anchor. There are a number of supertags for each lexeme to account for the different syntactic transformations (relative clause, *wh*-question, passivization etc.). For example, the verb *give* will be associated with at least these two trees, which we will call tdi and tdi-dat. (There are also many other trees.)



Supertagging is the task of disambiguating among the set of supertags associated with each word in a sentence, given the context of the sentence. In order to arrive at a complete parse, the only step remaining after supertagging is establishing the attachments among the supertags. Hence the result of supertagging is termed as an "almost parse" (Bangalore and Joshi, 1999).

The set of supertags is derived from the Penn Treebank using the approach of Chen (2001). This extraction procedure results in a supertag set of 4,727 supertags and about one million words of supertag annotated corpus. We use 950,028 annotated words for training (Sections 02-21) and 46,451 (Section 00) annotated words for testing in our experiments. We estimate the probability of a tag sequence directly as in discriminative classification approaches. In such approaches, the context of the word being supertagged is encoded as features for the classifier. Given the large scale multiclass labeling nature of the supertagging task, we train supertagging models as one-vs-rest binary classification problems. Detailed supertagging experiment results are reported in (Bangalore et al., 2005) which we summarize here. We use the lexical, part-of-speech attributes from the left and right context in a 6-word window and the lexical, orthographic (e.g. capitalization, prefix, suffix, digit) and part-of-speech attributes of the word being supertagged. Crucially, this set does not use the supertags for the words in the history. Thus during decoding the supertag assignment is done locally and does not need a dynamic programming search. We trained a Maxent model with such features using the labeled data set mentioned above and achieve an error rate of 11.48% on the test set.

3 Grammars and Models

MICA grammars are extracted in a three steps process. In a first step, a Tree Insertion Grammar (TIG) (Schabes and Waters, 1995) is extracted from the treebank, along with a table of counts. This is the grammar that is used for supertagging, as described in Section 2. In a second step, the TIG and the count table are used to build a PCFG. During the last step, the PCFG is "specialized" in order to model more finely some lexico-syntactic phenomena. The second and third steps are discussed in this section.

The extracted TIG is transformed into a PCFG which generates strings of supertags as follows. Initial elementary trees (which are substituted) yield rules whose left hand side is the root category of the elementary tree. Left (respectively right) auxiliary trees (the trees for which the foot node is the

left (resp. right) daughter of the root) give birth to rules whose left-hand side is of the form X_{l} (resp. X_{r}), where X is the root category of the elementary tree. The right hand side of each rule is built during a top down traversal of the corresponding elementary tree. For every node of the tree visited, a new symbol is added to the right hand side of rule, from left to right, as follows:

• The anchor of the elementary tree adds the supertag (i.e., the name of the tree), which is a terminal symbol, to the context-free rule.

• A substitution node in the elementary tree adds its nonterminal symbol to the context-free rule.

• A interior node in the elementary tree at which adjunction may occur adds to the context-free rule the nonterminal symbol X_{r}^* or X_{l}^*, where X is the node's nonterminal symbol, and l (resp. r) indicates whether it is a left (resp. right) adjunction. Each interior node is visited twice, the first time from the left, and then from the right. A set of non-lexicalized rules (i.e., rules that do not generate a terminal symbol) allow us to generate zero or more trees anchored by X_{l} from the symbol X_{l}^*. No adjunction, the first adjunction, and the second adjunction are modeled explicitly in the grammar and the associated probabilistic model, while the third and all subsequent adjunctions are modeled together.

This conversion method is basically the same as that presented in (Schabes and Waters, 1995), except that our PCFG models multiple adjunctions at the same node by positions (a concern Schabes and Waters (1995) do not share, of course). Our PCFG construction differs from that of Hwa (2001) in that she does not allow multiple adjunction at one node (Schabes and Shieber, 1994) (which we do since we are interested in the derivation structure as a representation of linguistic dependency). For more information about the positional model of adjunction and a discussion of an alternate model, the "bigram model", see (Nasr and Rambow, 2006).

Tree tdi from Section 2 gives rise to the following rule (where tdi and tCO are terminal symbols and the rest are nonterminals): $\mathrm{S} \rightarrow \mathrm{S}_{\mathrm{l}}^*$ NP $\mathrm{VP}_{\mathrm{l}}^*$ $\mathrm{V}_{\mathrm{l}}^*$ tdi $\mathrm{V}_{\mathrm{r}}^*$ NP $\mathrm{PP}_{\mathrm{l}}^*$ $\mathrm{P}_{\mathrm{l}}^*$ tCO $\mathrm{P}_{\mathrm{r}}^*$ NP $\mathrm{PP}_{\mathrm{r}}^*$ $\mathrm{VP}_{\mathrm{r}}^*$ $\mathrm{S}_{\mathrm{r}}^*$

The probabilities of the PCFG rules are estimated using maximum likelihood. The probabilistic model refers only to supertag names, not to words. In the basic model, the probability of the adjunction or substitution of an elementary tree (the daughter) in another elementary tree (the mother) only depends on the nonterminal, and does not depend on the mother nor on the node on which the attachment is performed in the mother elementary tree. It is well known that such a dependency is important for an adequate probabilistic modelling of syntax. In order to introduce such a dependency, we condition an attachment on the mother and on the node on which the attachment is performed, an operation that we call mother specialization. Mother specialization is performed by adding to all nonterminals the name of the mother and the address of a node. The specialization of a grammar increase vastly the number of symbols and rules and provoke severe data sparseness problems, this is why only a subset of the symbols are specialized.

4 Parser

SYNTAX (Boullier and Deschamp, 1988) is a system used to generate lexical and syntactic analyzers (parsers) (both deterministic and non-deterministic) for all kind of context-free grammars (CFGs) as well as some classes of contextual grammars. It has been under development at INRIA for several decades. SYNTAX handles most classes of deterministic (unambiguous) grammars (LR, LALR, RLR) as well as general context-free grammars. The non-deterministic features include, among others, an Earley-like parser generator used for natural language processing (Boullier, 2003).

Like most SYNTAX Earley-like parsers, the architecture of MICA's PCFG-based parser is the following:

• The Earley-like parser proper computes a shared parse forest that represents in a factorized (polynomial) way *all* possible parse trees according to the underlying (non-probabilistic) CFG that represents the TIG;

• Filtering and/or decoration modules are applied on the shared parse forest; in MICA's case, an n-best module is applied, followed by a dependency extractor that relies on the TIG structure of the CFG.

The Earley-like parser relies on Earley's algorithm (Earley, 1970). However, several optimizations have been applied, including guiding techniques (Boullier, 2003), extensive static (offline)

computations over the grammar, and efficient data structures. Moreover, Earley's algorithm has been extended so as to handle input DAGs (and not only sequences of forms). A particular effort has been made to handle huge grammars (over 1 million symbol occurrences in the grammar), thanks to advanced dynamic lexicalization techniques (Boullier and Sagot, 2007). The resulting efficiency is satisfying: with standard ambiguous NLP grammars, huge shared parse forest (over 10^{10} trees) are often generated in a few dozens of milliseconds.

Within MICA, the first module that is applied on top of the shared parse forest is SYNTAX's n-best module. This module adapts and implements the algorithm of (Huang and Chiang, 2005) for efficient n-best trees extraction from a shared parse forest. In practice, and within the current version of MICA, this module is usually used with $n = 1$, which identifies the optimal tree w.r.t. the probabilistic model embedded in the original PCFG; other values can also be used. Once the n-best trees have been extracted, the dependency extractor module transforms each of these trees into a dependency tree, by exploiting the fact that the CFG used for parsing has been built from a TIG.

5 Evaluation

We compare MICA to the MALT parser. Both parsers are trained on sections 02-21 of our dependency version of the WSJ PennTreebank, and tested on Section 00, not counting true punctuation. "Predicted" refers to tags (PTB-tagset POS and supertags) predicted by our taggers; "Gold" refers to the gold POS and supertags. We tested MALT using only POS tags (MALT-POS), and POS tags as well as 1-best supertags (MALT-all). We provide unlabeled ("Un") and labeled ("Lb") dependency accuracy (%). As we can see, the predicted supertags do not help MALT. MALT is significantly slower than MICA, running at about 30 words a second (MICA: 450 words a second).

	MICA		MALT-POS		MALT-all	
	Pred	Gold	Pred	Gold	Pred	Gold
Lb	85.8	97.3	86.9	87.4	86.8	96.9
Un	87.6	97.6	88.9	89.3	88.5	97.2

References

Srinivas Bangalore and Aravind Joshi. 1999. Supertagging: An approach to almost parsing. *Computational Linguistics*, 25(2):237–266.

Srinivas Bangalore, Patrick Haffner, and Gaël Emami. 2005. Factoring global inference by enriching local representations. Technical report, AT&T Labs – Reserach.

Pierre Boullier and Philippe Deschamp. 1988. Le système SYNTAX™ – manuel d'utilisation et de mise en œuvre sous UNIX™. http://syntax.gforge.inria.fr/syntax3.8-manual.pdf.

Pierre Boullier and Benoît Sagot. 2007. Are very large grammars computationnaly tractable? In *Proceedings of IWPT'07*, Prague, Czech Republic.

Pierre Boullier. 2003. Guided Earley parsing. In *Proceedings of the 7th International Workshop on =20 Parsing Technologies*, pages 43–54, Nancy, France.

John Chen. 2001. *Towards Efficient Statistical Parsing Using Lexicalized Grammatical Information*. Ph.D. thesis, University of Delaware.

Stephen Clark and James R. Curran. 2004. Parsing the WSJ using CCG and log-linear models. In *ACL'04*.

Jay Earley. 1970. An efficient context-free parsing algorithm. *Communication of the ACM*, 13(2):94–102.

Liang Huang and David Chiang. 2005. Better k-best parsing. In *Proceedings of IWPT'05*, Vancouver, Canada.

Rebecca Hwa. 2001. *Learning Probabilistic Lexicalized Grammars for Natural Language Processing*. Ph.D. thesis, Harvard University.

Aravind K. Joshi. 1987. An introduction to Tree Adjoining Grammars. In A. Manaster-Ramer, editor, *Mathematics of Language*. John Benjamins, Amsterdam.

Roger Levy and Christopher Manning. 2004. Deep dependencies from context-free statistical parsers: Correcting the surface dependency approximation. In *ACL'04*.

Dekang Lin. 1994. PRINCIPAR—an efficient, broad-coverage, principle-based parser. In *Coling'94*.

Alexis Nasr and Owen Rambow. 2006. Parsing with lexicalized probabilistic recursive transition networks. In *Finite-State Methods and Natural Language Processing*, Springer Verlag Lecture Notes in Commputer Science.

Joakim Nivre, Johan Hall, and Jens Nilsson. 2004. Memory-based dependency parsing. In *CoNLL-2004*.

Yves Schabes and Stuart Shieber. 1994. An alternative conception of tree-adjoining derivation. *Computational Linguistics*, 1(20):91–124.

Yves Schabes and Richard C. Waters. 1995. Tree Insertion Grammar. *Computational Linguistics*, 21(4).

Libin Shen and Aravind Joshi. 2005. Incremental ltag parsing. In *HLT-EMNLP'05*.

Wen Wang and Mary P. Harper. 2004. A statistical constraint dependency grammar (CDG) parser. In *Proceedings of the ACL Workshop on Incremental Parsing*.

Lexical and Syntactic Priming and Their Impact in Deployed Spoken Dialog Systems

Svetlana Stoyanchev and **Amanda Stent**
Department of Computer Science
Stony Brook University
Stony Brook, NY 11794-4400, USA
svetastenchikova@gmail.com, amanda.stent@stonybrook.edu

Abstract

In this paper, we examine user adaptation to the system's lexical and syntactic choices in the context of the deployed *Let's Go!* dialog system. We show that in deployed dialog systems with real users, as in laboratory experiments, users adapt to the system's lexical and syntactic choices. We also show that the system's lexical and syntactic choices, and consequent user adaptation, can have an impact on recognition of task-related concepts. This means that system prompt formulation, even in *flexible input* dialog systems, can be used to guide users into producing utterances conducive to task success.

1 Introduction

Numerous studies have shown that people adapt their syntactic and lexical choices in conversation to those of their conversational partners, both human (Brennan, 1996; Pickering et al., 2000; Lockridge and Brennan, 2002; Reitter et al., 2006) and computer (Branigan et al., 2003; Brennan, 1991; Brennan, 1996; Gustafson et al., 1997; Ward and Litman, 2007). User adaptation to the system's lexical and syntactic choices can be particularly useful in *flexible input* dialog systems. *Limited input* dialog systems, including most commercial systems, require the user to respond to each system prompt using only the concept and words currently requested by the system. *Flexible input* dialog systems allow the user to respond to system prompts with concepts and words in addition to or other than the ones currently requested, and may even allow the user to take task initiative. Speech recognition (ASR) accuracy in *limited input* systems is better than in *flexible input* systems (Danieli and Gerbino, 1995; Smith and Gordon, 1997). However, task completion rates and times are better in *flexible input* systems (Chu-Carroll and Nickerson, 2000; Smith and Gordon, 1997). With user adaptation, in *flexible input* dialog systems prompts can be formulated to maximize ASR accuracy and reduce the number of ASR timeouts (Sheeder and Balogh, 2003).

Previous research on user adaptation to dialog systems was conducted in laboratory settings. However, the behavior of recruited subjects in a quiet laboratory may differ from that of real users in the noisy world (Ai et al., 2007). Here we present the first study, to the best of our knowledge, that investigates the adaptive behavior of real users of a live dialog system. We analyze dialogs from CMU's *Let's Go!* dialog system (Raux et al., 2005). We look at the effects of the system's lexical and syntactic choices on: 1) lexical and syntactic choices in user responses; and 2) concept identification rates for user responses. We confirm prior results showing that users adapt to the system's lexical and syntactic choices. We also show that particular choices for system prompts can lead to higher concept identification rates.

2 Experimental Method

We conducted our experiment using the *Let's Go!* telephone-based spoken dialog system that provides information about bus routes in Pittsburgh (Raux et al., 2005). The users are naive callers from the general population seeking information about bus

Proceedings of NAACL HLT 2009: Short Papers, pages 189–192,
Boulder, Colorado, June 2009.

condition	request departure location	confirm departure location	request arrival location	confirm arrival location
(1)	Where are you **leaving from**?	**Leaving from X**, is this correct?	Where are you **going to**?	**Going to** X, is this correct
(2)	Where are you **leaving from**?	**From** X, is this correct?	Where are you **going to**?	**To** X, is this correct
(3)	What is the place of your departure	X, is this correct?	What is the place of your arrival?	X, is this correct
(4)	Where do you want to **leave from**?	You want to **leave from** X, is this correct?	Where do you want to **go to**?	You want to **go to X**, is this correct

Table 1: Experimental conditions

Spkr	Task type	Utterance
Sys	Open	Welcome to the CMU Let's Go bus information system. What can I do for you?
Usr		*61A schedule*
Sys	Request Departure	Where do you wanna leave from?
Usr	Location	*From downtown*
Sys	Confirm Departure	Leaving from downtown. Is this correct?
Usr	Location	*Yes*
Sys	Request Arrival	Where are you going to?
Usr	Location	*Oakland*
Sys	Confirm Arrival	Going to Waterfront. Is this correct?
Usr	Location	*No, to Oakland*

Figure 1: Dialog extract from *Let's Go!* data

schedules. In order to provide the user with route information, *Let's Go!* elicits a departure location, a destination, a departure time, and optionally a bus route number. Each concept value provided by the user is explicitly confirmed by the system. Figure 1 shows an example dialog with the system.

Let's Go! is a *flexible input* dialog system. The user can respond to a system prompt using a single word or short phrase, e.g. *Downtown*, or a complete sentence, e.g. *I am leaving from downtown*[1].

We ran four experimental conditions for two months. The conditions varied in the lexical choice and syntax of system prompts for two system *request location* tasks and two system *confirm location* tasks (see Table 1). System prompts differed by presence of a verb (*to leave, to go*) or a preposition (*to, from*), and by the syntactic form of the verb. The *request location* prompt contained both a verb and a preposition in the experimental conditions (1, 3, and 4). The *confirm location* prompt contained both a verb and a preposition in conditions 1 and 4, only a preposition in condition 2, and neither verb nor preposition in condition 3. In conditions 1 and 4, both request and confirmation prompts differed in the verb form (*leaving/leave, going/go*).

2184 dialogs were used for this analysis. For each experimental condition, we counted the percentages of verbs, verb forms, prepositions, and locations in the ASR output for user responses to system *request location* and *confirm location* prompts. Although the data contains recognition errors, the only difference in system functionality between the conditions is the formulation of the system prompt, so any statistically significant difference in user responses between different conditions can be attributed to the formulation of the prompt.

3 Syntactic Adaptation

We analyze whether users are more likely to use action verbs (*leave, leaving, go,* or *going*) and prepositions (*to, from*) in response to system prompts that use a verb or a preposition. This analysis is interesting because ASR partially relies on *context words*, words related to a particular concept type such as place, time or bus route. For example, the likelihood of correctly recognizing the location *Oakland* in the utterance *"going to Oakland"* is different from the likelihood of correctly recognizing the single word utterance *"Oakland"*.

Table 2 shows the percentages of user responses

[1]The user response can also contain concepts not requested in the prompt, e.g. specifying departure location and bus number in one response.

Cond.	Sys uses verb	Sys uses prep	% with verb	% with prep
Responses to *request location* prompt				
(1)	yes	yes	2.3% ∗	5.6%
(2)	yes	yes	1.9%	4.3%
(3)	**no**	**no**	**0.7%**	4.5%
(4)	yes	yes	2.4%∗	6.0%
Responses to *confirm location* prompt				
(1)	yes	yes	15.7% ∗ ♠	23.4%
(2)	**no**	yes	**3.9%**	16.9%
(3)	**no**	**no**	**6.4%**	12.7%
(4)	yes	yes	10.8%	22.0%

Table 2: Percentages of user utterances containing verbs and prepositions. ∗ indicates a statistically significant difference ($p<0.01$) from the *no action verb* condition (3). ♠ indicates a statistically significant difference from the *no action verb in confirmation* condition (2).

in each experimental condition that contain a verb and/or a preposition. We observe adaptation to the presence of a verb in user responses to *request location* prompts. The prompts in conditions 1, 2 and 4 contain a verb, while those in condition 3 do not. The differences between conditions 1 and 3, and between conditions 4 and 3, are statistically significant ($p<0.01$)[2]. The difference between conditions 2 and 3 is not statistically significant, perhaps due to the absence of a verb in a prior *confirm location* prompt.

A similar adaptation to the presence of a verb in the system prompt is seen in user responses to *confirm location* prompts. The prompts in conditions 1 and 4 contain a verb while those in conditions 2 and 3 do not. The differences between conditions 1 and 2, and between conditions 1 and 3, are statistically significant ($p<.01$), while the difference between conditions 4 and 2 exhibits a trend. We hypothesize that the lack of the statistically significant differences between conditions 4 and 2, and conditions 4 and 3, is caused by the low relative frequency in our data of dialogs in condition 4.

We do not find statistically significant differences in the use of prepositions. However, we observe a trend showing higher likelihood of a preposition in user responses to *confirm location* in the conditions where the system uses a preposition. Prepositions are short closed-class context words that are more likely to be misrecognized (Goldwater et al., 2008).

[2]All analyses in this section are t-tests with Bonferroni adjustment.

Condition/ User's verb	LEAVING (progressive)	LEAVE (simple)	total
(1) Progressive	74.5%	25.5%	55
(3) Neutral	61.3%	38.7%	31
(4) Simple	43%	57%	42
Condition/ User's verb	GOING (progressive)	GO (simple)	total
(1) Progressive	84.4%	15.6%	45
(3) Neutral	66.6%	33.4%	21
(4) Simple	46.5%	53.5%	43

Table 3: Usage of verb forms in user utterances

Hence, more data (or human transcription) may be required to see a statistically significant effect.

4 Lexical Adaptation

We analyze whether system choice of a particular verb form affects user choice of verb form. For this analysis we only consider user utterances in response to a *request location* or *confirm location* prompt that contain a concept and at least one of the verb forms *leaving, going, leave,* or *go*[3].

Table 3 shows the total counts and percentages of each verb form in the *progressive form* condition (condition 1), and the *neutral* condition (condition 3), and the *simple form* condition (condition 4)[4]. We find that the system's choice of verb form has a statistically significant impact on the user's choice (χ^2 test, $p<0.01$). In the *neutral* condition, users are more likely to choose the progressive verb form. In the *progressive form* condition, this preference increases by 13.2% for the verb *to leave,* and by 17.8% for the verb *to go*. By contrast, in the *simple form* condition, this preference decreases by 18.3% for the verb *to leave* and by 20.1% for the verb *to go,* making users slightly more likely to choose the simple verb form than the progressive verb form.

5 Effect of Adaptation on Speech Recognition Performance

The correct identification and recognition of task-related concepts in user utterances is an essential functionality of a dialog system. Table 4 shows

[3]Such utterances constitute 3% of all user responses to all *request* and *confirm place* prompts in our data.

[4]We ignore condition 2 where the verb is used only in the *request* prompt.

System prompt	Arrival request	Departure request
(1)	72.2% ∗	63.8%
(2)	77.4%	61.0%
(3)	74.5% ∗	61.5%
(4)	**82.0%**	66.0%

Table 4: Concept identification rates following *request location* prompts. ∗ indicates a statistically significant difference ($p<0.01$ with Bonferroni adjustment) from condition 4.

the percentage of user utterances following a *request location* prompt that contain an automatically-recognized location concept. Condition 4, where the system prompt uses the verb form *to leave*, achieves the highest concept identification rates. The differences in concept identification rates between conditions 1 and 4, and between conditions 3 and 4, are statistically significant for *request arrival location* (t-test, $p<.01$). Other differences are not statistically significant, perhaps due to lack of data.

6 Conclusions and Future Work

In this paper, we showed that in deployed dialog systems with real users, as in laboratory experiments, users adapt to the lexical and syntactic choices of the system. We also showed that user adaptation to system prompts can have an impact on recognition of task-related concepts. This means that the formulation of system prompts, even in *flexible input* dialog systems, can be used to guide users into producing utterances conducive to task success.

In future work, we plan to confirm these results using transcribed data. We also plan additional experiments on adaptation in *Let's Go!*, including an analysis of the time course of adaptation and further analyses of the impact of adaptation on ASR performance.

7 Acknowledgements

We would like to thank the *Let's Go!* researchers at CMU for making *Let's Go!* available. This research was supported by the NSF under grant no. 0325188.

References

H. Ai, A. Raux, D. Bohus, M. Eskenazi, and D. Litman. 2007. Comparing spoken dialog corpora collected with recruited subjects versus real users. In *Proceedings of SIGDial*.

H. Branigan, M. Pickering, J. Pearson, J. McLean, and C. Nass. 2003. Syntactic alignment between computers and people: the role of belief about mental states. In *Proceedings of CogSci*.

S. Brennan. 1991. Conversation with and through computers. *User Modeling and User-Adapted Interaction*, 1(1):67–86.

S. Brennan. 1996. Lexical entrainment in spontaneous dialog. In *Proceedings of ISSD*.

J. Chu-Carroll and J. Nickerson. 2000. Evaluating automatic dialogue strategy adaptation for a spoken dialogue system. In *Proceedings of NAACL*.

M. Danieli and E. Gerbino. 1995. Metrics for evaluating dialogue strategies in a spoken language system. In *Proceedings of the AAAI Spring Symposium on Empirical Methods in Discourse Interpretation and Generation*.

S. Goldwater, D. Jurafsky, and C. Manning. 2008. Which words are hard to recognize? Lexical, prosodic, and disfluency factors that increase asr error rates. In *Proceedings of ACL/HLT*.

J. Gustafson, A. Larsson, R. Carlson, and K. Hellman. 1997. How do system questions influence lexical choices in user answers? In *Proceedings of Eurospeech*.

C. Lockridge and S. Brennan. 2002. Addressees' needs influence speakers' early syntactic choices. *Psychonomics Bulletin and Review*.

M. Pickering, H. Branigan, A. Cleland, and A. Stewart. 2000. Activation of syntactic priming during language production. *Journal of Psycholinguistic Research*, 29(2):205–216.

A. Raux, B. Langner, A. Black, and M Eskenazi. 2005. Let's Go public! taking a spoken dialog system to the real world. In *Proceedings of Eurospeech*.

E. Reitter, J. Moore, and F. Keller. 2006. Priming of syntactic rules in task-oriented dialogue and spontaneous conversation. In *Proceedings of CogSci*.

T. Sheeder and J. Balogh. 2003. Say it like you mean it: priming for structure in caller responses to a spoken dialog system. *International Journal of Speech Technology*, 6(2):103–111.

R. Smith and S. Gordon. 1997. Effects of variable initiative on linguistic behavior in human-computer spoken natural language dialogue. *Computational Linguistics*, 23(1):141–168.

A. Ward and D. Litman. 2007. Automatically measuring lexical and acoustic/prosodic convergence in tutorial dialog corpora. In *Proceedings of the SLaTE Workshop on Speech and Language Technology in Education*.

Analysing Recognition Errors in Unlimited-Vocabulary Speech Recognition

Teemu Hirsimäki and **Mikko Kurimo**
Adaptive Informatics Research Centre
Helsinki University of Technology
P.O. Box 5400, 02015, TKK, Finland
teemu.hirsimaki@tkk.fi

Abstract

We analyze the recognition errors made by a morph-based continuous speech recognition system, which practically allows an unlimited vocabulary. Examining the role of the acoustic and language models in erroneous regions shows how speaker adaptive training (SAT) and discriminative training with minimum phone frame error (MPFE) criterion decrease errors in different error classes. Analyzing the errors with respect to word frequencies and manually classified error types reveals the most potential areas for improving the system.

1 Introduction

Large vocabulary speech recognizers have become very complex. Understanding how the parts of the system affect the results separately or together is far from trivial. Still, analyzing the recognition errors may suggest how to reduce the errors further.

There exist previous work on analyzing recognition errors. Chase (1997) developed error region analysis (ERA), which reveals whether the errors are due to acoustic or language models. Greenberg et al. (2000) analyzed errors made by eight recognition systems on the Switchboard corpus. The errors correlated with the phone misclassification and speech rate, and conclusion was that the acoustic front ends should be improved further. Duta et al. (2006) analyzed the main errors made by the 2004 BBN speech recognition system. They showed that errors typically occur in clusters and differ between broadcast news (BN) and conversational telephone speech (CTS) domains. Named entities were a common cause for errors in the BN domain, and hesitation, repeats and partially spoken words in the CTS domain.

This paper analyzes the errors made by a Finnish morph-based continuous recognition system (Hirsimäki et al., 2009). In addition to partitioning the errors using ERA, we compare the number of letter errors in different regions and analyze what kind of errors are corrected when speaker adaptive training and discriminative training are taken in use. The most potential error sources are also studied by partitioning the errors according to manual error classes and word frequencies.

2 Data and Recognition System

The language model training data used in the experiments consist of 150 million words from the Finnish Kielipankki corpus. Before training the n-gram models, the words of the training data were split into morphs using the Morfessor algorithm, which has been shown to improve Finnish speech recognition (Hirsimäki et al., 2006). The resulting morph lexicon contains 50 000 distinct morphs. A growing algorithm (Siivola et al., 2007) was used for training a Kneser-Ney smoothed high-order variable-length n-gram model containing 52 million n-grams.

The acoustic phoneme models were trained on the Finnish SpeechDat telephone speech database: 39 hours from 3838 speakers for training, 46 minutes from 79 speakers for development and another similar set for evaluation. Only full sentences were used and sentences with severe noise or mispronunciations were removed.

Proceedings of NAACL HLT 2009: Short Papers, pages 193–196,
Boulder, Colorado, June 2009.

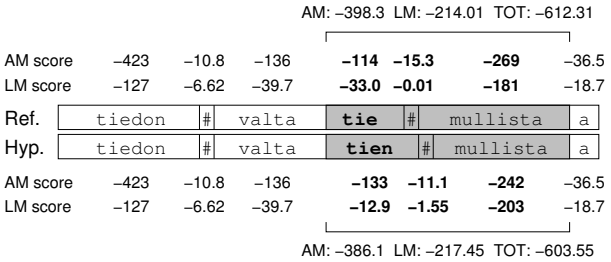


Figure 1: An example of a HYP-AM error region. The scores are log probabilities. Word boundaries are denoted by '#'. The error region only contains one letter error (an inserted 'n').

	Letter errors		
Region	ML	ML+SAT	ML+SAT+MPFE
HYP-BOTH	962	909	783
HYP-AM	1059	709	727
HYP-LM	623	597	425
REF-TOT	82	60	15
Total	2726	2275	1950
LER (%)	6.8	5.6	4.8

Table 1: SpeechDat: Letter errors for different training methods and error regions. The reference transcript contains 40355 letters in total.

The acoustic front-end consist of 39-dimensional feature vectors (Mel-frequency cepstral coefficients with first and second time-derivatives), global maximum likelihood linear transform, decision-tree tied HMM triphones with Gaussian mixture models, and cepstral mean subtraction.

Three models are trained: The first one is a *maximum likelihood* (ML) model without any adaptation. The second model (ML+SAT) enhances the ML model with three iterations of *speaker adaptive training* (SAT) using *constrained maximum likelihood linear regression* (CMLLR) (Gales, 1998). In recognition, unsupervised adaptation is applied in the second pass. The third model (ML+SAT+MPFE) adds four iterations of discriminative training with *minimum phone frame error* (MPFE) criterion (Zheng and Stolcke, 2005) to the ML+SAT model.

3 Analysis

3.1 Error Region Analysis

Error Region Analysis (Chase, 1997) can be used to find out whether the language model (LM), the acoustic model (AM) or both can be blamed for an erroneous region in the recognition output. Figure 1 illustrates the procedure. For each utterance, the final hypothesis is compared to the forced alignment of the reference transcript and segmented into correct and error regions. An *error region* is a contiguous sequence of morphs that differ from the corresponding reference morphs with respect to morph identity, boundary time-stamps, AM score, LM score, or n-gram history[1].

By comparing the AM and LM scores in the hypothesis and reference regions, the regions can be divided in classes. We denote the recognition hypothesis as HYP, and the reference transcript as REF. The relevant classes for the analysis are the following. REF-TOT: the reference would have better total score, but it has been erroneously pruned. HYP-AM: the hypothesis has better score, but only AM favors HYP over REF. HYP-LM: the hypothesis has better score, but only LM favors HYP over REF. HYP-BOTH: both the AM and LM favor HYP.

Since the error regions are independent, the letter error rate[2] (LER) can be computed separately for each region. Table 1 shows the error rates for three different acoustic models: ML training, ML+SAT, and ML+SAT+MPFE. We see that SAT decreases all error types, but the biggest reduction is in the HYP-AM class. This should be expected. In the ML case, the Gaussian mixtures contain much variance due to different unnormalized speakers, and since the test set contains only unseen speakers, many errors are expected for some speakers. Adapting the models to the test set is expected to increase the acoustic score of the reference transcript, and since in the HYP-AM regions the LM already prefers REF, corrections because of SAT are most probable there.

On the other hand, adding MPFE after SAT seems

[1] A region may be defined as an error region even if the transcription is correct (only the segmentation differs). However, since we are going to analyze the number of letter errors in the error regions, the "correct" error regions do not matter.

[2] The words in Finnish are often long and consist of several morphs, so the performance is measured in letter errors instead of word errors to have finer resolution for the results.

	Letter errors					
Class label	Total	HYP-BOTH	HYP-AM	HYP-LM	REF-TOT	Class description
Foreign	156	89	**61**	6		Foreign proper name
Inflect	143	74	26	**43**		Small error in inflection
Poor	131	37	**84**		10	Poor pronunciation or repair
Noise	124	21	**97**	6		Error segment contains some noise
Name	81	29	**29**	23		Finnish proper name
Delete	65	29	9	**27**		Small word missing
Acronym	53	44	**6**	3		Acronym
Compound	42	11	8	**23**		Word boundary missing or inserted
Correct	37	15	**19**	3		Hypothesis can be considered correct
Rare	27	11	3	**13**		Reference contains a very rare word
Insert	9	3	**6**			Small word inserted incorrectly
Other	1082	421	**379**	277	5	Other error

Table 2: Manual error classes and the number of letter errors for the ML+SAT+MPFE system.

to reduce HYP-BOTH and HYP-LM errors, but not HYP-AM errors. The number of search errors (REF-TOT) also decreases.

All in all, for all models, there seems to be more HYP-AM errors than HYP-LM errors. Chase (1997) lists the following possible reasons for the HYP-AM regions: noise, speaker pronounces badly, pronunciation model is poor, some phoneme models not trained to discriminate, or reference is plainly wrong. The next section studies these issues further.

3.2 Manual Error Classification

Next, the letter errors in the error regions were manually classified according to the most probable cause. Table 2 shows the classes, the total number of letter errors for each class, and the errors divided to different error region types.

All errors that did not seem to have an obvious cause are put under the class *Other*. Some of the errors were a bit surprising, since the quality of the audio and language seemed perfectly normal, but still the recognizer got the sentences wrong. On the other hand, the class also contains regions where the speech is very fast or the signal level is quite low.

The largest class with a specific cause is *Foreign*, which contains about 8 % of all letter errors. Currently, the morph based recognizer does not have any foreign pronunciation modeling, so it is natural that words like *Ching, Yem Yung, Villeneuve, Schumacher, Direct TV, Thunderbayssa* are not recognized correctly, since the mapping between the written form and pronunciation does not follow the normal Finnish convention. In Table 2 we see, that the acoustic model prefers the incorrect hypothesis in almost all cases. A better pronunciation model would be essential to improve the recognition. However, integrating exceptions in pronunciation to morph-based recognition is not completely straightforward. Another difficulty with foreign names is that they are often rare words, so they will get low language model probability anyway.

The errors in the *Acronym* class are pretty much similar to foreign names. Since the letter-by-letter pronunciation is not modelled, the acronyms usually cause errors.

The next largest class is *Inflect*, which contains errors where the root of the word is correctly recognized, but the inflectional form is slightly wrong (for example: *autolla/autolle, kirjeeksi/kirjeiksi*). In these errors, it is usually the language model that prefers the erroneous hypothesis.

The most difficult classes to improve are perhaps *Poor* and *Noise*. For bad pronunciations and repairs it is not even clear what the correct answer should be. Should it be the word the speaker tried to say, or the word that was actually said? As expected, the language model would have preferred the correct hypothesis in most cases, but the acoustic model have chosen the wrong hypothesis.

The *Name* and *Rare* are also difficult classes. Contrary to the foreign names and acronyms, the pronunciation model is not a problem.

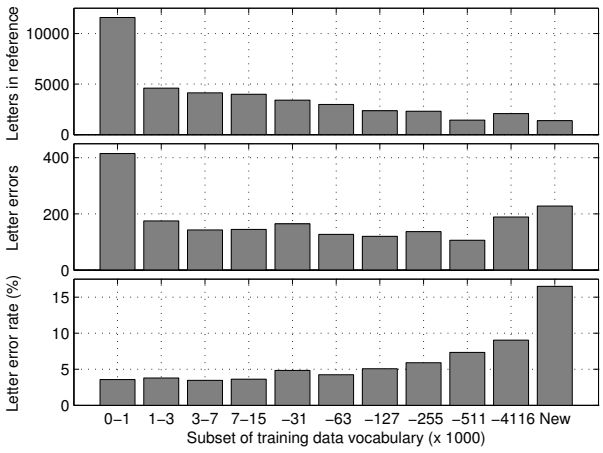


Figure 2: Frequency analysis of the SAT+MPFE system. Number of letters in reference (top), number of letter errors (middle), and letter error rate (bottom) partitioned according to word frequencies. The leftmost bar corresponds to the 1000 most frequent words, the next bar to the 2000 next frequent words, and so on. The rightmost bar corresponds to words not present in the training data.

The *Compound* errors are mainly in HYP-LM regions, which is natural since there is usually little acoustic evidence at the word boundary. Furthermore, it is sometimes difficult even for humans to know if two words are written together or not. Sometimes the recognizer made a compound word error because the compound word was often written incorrectly in the language model training data.

3.3 Frequency Analysis

In order to study the effect of rare words in more detail, the words in the test data were grouped according their frequencies in the LM training data: The first group contained all the words that were among the 1000 most common words, the next group contained the next 2000 words, then 4000, and so on, until the final group contained all words not present in the training data.

Figure 2 shows the number of letters in the reference (top), number of letter errors (middle), and letter error rate (bottom) for each group. Quite expectedly, the error rates (bottom) rise steadily for the infrequent words and is highest for the new words that were not seen in the training data. But looking at the absolute number of letter errors (middle), the majority occur in the 1000 most frequent words.

4 Conclusions

SAT and MPFE training seem to correct different error regions: SAT helps when the acoustic model dominates and MPFE elsewhere. The manual error classification suggests that improving the pronunciation modeling of foreign words and acronyms is a potential area for improvement. The frequency analysis shows that a major part of the recognition errors occur still in the 1000 most common words. One solution might be to develop methods for detecting when the problem is in acoustics and to trust the language model more in these regions.

Acknowledgments

This work was partly funded from the EC's FP7 project EMIME (213845).

References

Lin Chase. 1997. *Error-Responsive Feedback Mechanisms for Speech Recognizers*. Ph.D. thesis, Robotics Institute, Carnegie Mellon University.

Nicolae Duta, Richard Schwartz, and John Makhoul. 2006. Analysis of the errors produced by the 2004 BBN speech recognition system in the DARPA EARS evaluations. *IEEE Trans. Audio, Speech Lang. Process.*, 14(5):1745–1753.

M. J. F. Gales. 1998. Maximum likelihood linear transformations for HMM-based speech recognition. *Computer Speech and Language*, 12(2):75–98.

Steven Greenberg, Shuangyu Chang, and Joy Hollenback. 2000. An introduction to the diagnostic evaluation of the Switchboard-corpus automatic speech recognition systems. In *Proc. NIST Speech Transcription Workshop*.

Teemu Hirsimäki, Mathias Creutz, Vesa Siivola, Mikko Kurimo, Sami Virpioja, and Janne Pylkkönen. 2006. Unlimited vocabulary speech recognition with morph language models applied to Finnish. *Computer Speech and Language*, 20(4):515–541.

Teemu Hirsimäki, Janne Pylkkönen, and Mikko Kurimo. 2009. Importance of high-order n-gram models in morph-based speech recognition. *IEEE Trans. Audio, Speech Lang. Process.*, 17(4):724–732.

Vesa Siivola, Teemu Hirsimäki, and Sami Virpioja. 2007. On growing and pruning Kneser-Ney smoothed n-gram models. *IEEE Trans. Audio, Speech Lang. Process.*, 15(5):1617–1624.

Jing Zheng and Andreas Stolcke. 2005. Improved discriminative training using phone lattices. In *Proc. Interspeech*, pages 2125–2128.

The independence of dimensions in multidimensional dialogue act annotation

Volha Petukhova and **Harry Bunt**
Tilburg Center for Creative Computing
Tilburg University, The Netherlands,
{v.petukhova,h.bunt}@uvt.nl

Abstract

This paper presents empirical evidence for the orthogonality of the DIT^{++} multidimensional dialogue act annotation scheme, showing that the ten dimensions of communication which underlie this scheme are addressed independently in natural dialogue.

1 Introduction

Studies of human dialogue behaviour indicate that natural dialogue utterances are very often multifunctional. This observation has inspired the development of multidimensional approaches to dialogue analysis and annotation, e.g. (Allen & Core, 1997) , (Larsson, 1998), (Popescu-Belis, 2005), (Bunt, 2006). The most frequently used annotation scheme that implements this approach is DAMSL (Allen and Core, 1997), which allows multiple labels to be assigned to utterances in four layers: Communicative Status, Information Level, Forward-Looking Function (FLF) and Backward-Looking Function (BLF). The FLF layer is subdivided into five classes, including (roughly) the classes of commissive and directive functions, well known from speech act theory. The BLF layer has four classes: Agreement, Understanding, Answer, and Information Relation. These nine classes, also referred to as 'dimensions', form mutually exclusive sets of tags; no further motivation is given for the particular choice of classes.

Popescu-Belis (2005) argues that dialogue act tagsets should seek a multidimensional theoretical grounding and defines the following aspects of utterance function that could be relevant for choosing dimensions (1) the traditional clustering of illocutionary forces in speech act theory into five classes: Representatives, Commissives, Directives, Expressives and Declarations; (2) turn management; (3) adjacency pairs; (4) topical organization in dialogue; (5) politeness functions; and (6) rhetorical roles.

Structuring an annotation scheme by grouping related communicative functions into clusters makes the structure of the schema more transparent. Such clusters or 'dimensions' are usually defined as a set of functions related to the same type of information, such as Acknowledging, Signalling Understanding and Signalling Non-understanding, or Dialogue Opening and Dialogue Closing. Bunt (2006) shows that this does not always lead to a notion of dimension that has any conceptual and theoretical significance, and argues that some of the function classes of DAMSL do not constitute proper dimensions.

In particular, a theoretically grounded multidimensional schema should provide an account of the possible multifunctionality of dialogue utterances. In (Bunt, 2006); (Bunt and Girard, 2005) a dimension in dialogue act analysis is defined as *an aspect of participating in dialogue* which can be addressed:

- *by dialogue acts which have a function specifically for dealing with this aspect;*
- *independently of the other dimensions.*

The independence of dimensions, required by this definition, has the effect that an utterance may have a function in one dimension independent of the functions that it may have in other dimensions, and helps to explain why utterances may have multiple functions. Moreover, it leads to more manageable and

Proceedings of NAACL HLT 2009: Short Papers, pages 197–200,
Boulder, Colorado, June 2009.

more adaptable annotation schemas (compared to, for instance, DAMSL and its derivatives), since it allows annotators to leave out certain dimensions that they are not interested in, or to extend the schema with additional dimensions; and it allows restricting or modifying the set of tags in a particular dimension without affecting the rest of the schema.

Based on the above definition and extensive theoretical and empirical studies, 10 dimensions are defined in the DIT^{++} dialogue act annotation scheme[1]: the domain or task/activity (*Task*); feedback on the processing of previous utterances by the speaker (*Auto-feedback*) or by other interlocutors (*Allo-feedback*); managing difficulties in the speaker's utterance production (*Own-Communication Management, OCM*) or that of other interlocutors (*Partner Communication Management, PCM*); the speaker's need for time to continue the dialogue (*Time Management*); establishing and maintaining contact (*Contact Management*); the allocation of the next turn (*Turn Management*); the way the speaker is planning to structure the dialogue (*Dialogue Structuring*); and attention for social aspects of the interaction (*Social Obligations Management, SOM*).

This paper investigates the independence of these ten dimensions. In Section 2 we discuss the notion of independence of dimensions and how it can be tested. Section 3 reports test results and Section 4 draws conclusions.

2 Independence of dimensions

We define two dimensions D1 and D2 in an annotation scheme to be independent iff (1) an utterance may be assigned a value in D1 regardless of whether it is assigned a value in D2; and (2) it is not the case that whenever an utterance has a value in D1, this determines its value in D2.[2]

Dependences between dimensions can be determined empirically by analyzing annotated dialogue data. Dimension tags which always co-occur are nearly certainly dependent; zero co-occurrence scores also suggest possible dependences. Besides co-occurrence scores, we also provide a statistical analysis using the phi coefficient as a measure of relatedness. The phi measure is related to the chi-square statistic, used to test the independence of categorical variables, and is similar to the correlation coefficient in its interpretation.

If a dimension is not independent from other dimensions, then there would be no utterances in the data which address only that dimension. We therefore also investigate to which extent it happens that an utterance addresses only one dimension. We also investigate whether a dimension is addressed only in reaction to a certain other dimension. For example, the *answer* dimension as defined in DAMSL cannot be seen as independent, because *answers* need *questions* in order to exist. The test here is to examine the relative frequencies of pairs <dimension tag, previous dimension tag>.

To sum up, we performed four tests, examining:

1. the relative frequency of *communicative function co-occurrences* across dimensions;
2. *the extent of relatedness between dimensions* measure with the phi coefficient;
3. for all dimensions whether there are utterances *addressing only that dimension*;
4. the relative frequency of pairs of *dimension* and *previous dimension*.

3 Test results

Since different types of dialogue may have different tag distributions, three different dialogue corpora have been examined:

- The DIAMOND corpus[3] of two-party instructional human-human Dutch dialogues (1,408 utterances);
- The AMI corpus[4] of task-oriented human-human multi-party English dialogues (3,897 utterances);
- The OVIS corpus[5] of information-seeking human-computer Dutch dialogues (3,942 utterances).

All three corpora were manually segmented and tagged according to the DIT++ annotation scheme.

[1] For more information about the scheme and its dimensions please visit `http://dit.uvt.nl/`

[2] See Petukhova and Bunt (2009) for a more extensive discussion.

[3] For more information see Geertzen, J., Girard, Y., and Morante R. 2004. The DIAMOND project. Poster at CATALOG 2004.

[4] **A**ugmented **M**ulti-party **I**nteraction (`http://www.amiproject.org/`)

[5] **O**penbaar **V**ervoer **I**nformatie **S**ystem (Public Transport Information System) http://www.let.rug.nl/vannoord/Ovis/

within	Task	Auto-F.	Allo-F.	Turn M.	Time M.	DS	Contact M.	OCM	PCM	SOM
Task	-	0.05(67.9)	0(24.9)	10.2(97.5)	1.4(2.4)	1.4(1.5)	0 (0.4)	5.1(69.6)	0(0.1)	0(0.7)
Auto-F.	0.7(78.9)	-	0(0)	9.1(98.7)	0.6(1.4)	0.3(1.2)	0(20.2)	0(0.7)	0(65.0)	0(0.7)
Allo-F.	0(24.9)	0	-	59.2(94.8)	1.2(35.7)	0(2.1)	0(1.2)	0(7.9)	0.6(0.7)	0(0.3)
Turn M.	50.2(76.0)	3.5(66.2)	5.6(19.4)	-	8.0(42.9)	1.2(3.9)	0.1(13.8)	25(99.6)	0.2(1.0)	0.2(0.5)
Time M.	28.2(13.4)	0.5(11.3)	2.8(7.8)	96.9(98.6)	-	0.7(1.7)	0(0)	2.5(83.2)	0(0.5)	0(0)
DS	28.3(92.2)	0.4(58.3)	0(29.1)	22.6(87.5)	4.2(4.9)	-	0(25.0)	0(3.7)	0(0)	3.2(12.5)
Contact M.	0(2.4)	0(97.1)	0(1.6)	18.2(98.8)	0(0)	0(2.4)	-	0 (0.3)	0(0)	0(0)
OCM	75.5(82.2)	0(0.8)	0(2.5)	82.9(96.9)	3.4(7.8)	1.3(3.9)	0(13.5)	-	0 (0.9)	0.2(0.6)
PCM	0(11.8)	0(65.0)	4.9(11.8)	12.2(79.1)	0(12.2)	0(0)	0(0)	0(0)	-	0(0)
SOM	0(0.7)	0(80.0)	0(10.0)	6(90.0)	0(0)	10.0(30.0)	0(0)	0(2.0)	0(0)	-

Table 1: *Co-occurrences of communicative functions across dimensions in AMI corpus expressed in relative frequency in % implicated and entailed functions excluded and included (in brackets).*

The test results presented in this section are similar for all three corpora.

The co-occurrence results in Table 1 show no dependences between dimensions, although some combinations of dimensions occur frequently, e.g. time and turn management acts often co-occur. A speaker who wants to win some time to gather his thoughts and uses Stalling acts mostly wants to continue in the sender role, and his stalling behaviour may be intended to signal that as well (i.e., to be interpreted as a Turn Keeping act). But stalling behaviour does not always have that function; especially an extensive amount of stallings accompanied by relatively long pauses may be intended to elicit support for completing an utterance.

It is also interesting to have a look at co-occurrences of communicative functions taking implicated and entailed functions into account (the corpora were reannotated for this purpose). An implicated function is for instance the positive feedback (on understanding and evaluating the preceding utterance(s) of the addressee) that is implied by an expression of thanks; examples of entailed functions are the positive feedback on the preceding utterance that is implied by answering a question, by accepting an invitation, or by rejecting an offer.

Co-occurrence scores are higher when entailed and implicated functions are taken into account (the scores given in brackets in Table 1). For example, questions, which mostly belong to the Task dimension, much of the time have an accompanying Turn Management function, either releasing the turn or assigning it to another dialogue participant, allowing the question to be answered. Similarly, when accepting a request the speaker needs to have the turn, so communicative functions like Accept Request will often be accompanied by functions like Turn Take or Turn Accept. Such cases contribute to the co-occurrence score between the Task and Turn Management dimensions.

Table 1 shows that some dimensions do not occur in combination. We do not find combinations of Contact and Time Management, Contact and Partner Communication Management, or Partner Communication Management and Discourse Structuring, for example. Close inspection of the definitions of the tags in these pairs of dimensions does not reveal combination restrictions that would make one of these dimensions depend on the others.

Table 2 presents the extent to which dimensions are related when the corpus data are annotated with or without taking implicated and entailed functions into account, according to the calculated phi coefficient.

No strong positive (phi values from .7 to 1.0) or negative (-.7 to -1.0) relations are observed. There is a weak positive association (.6) between Turn and Time Management (see co-occurrence analysis above) and between OCM and Turn Management (.4). Weak negative associations are observed between Task and Auto-feedback (-.5) when entailed and implicated functions are not considered; between Task and Contact Management (-.6); and between Auto- and Allo-feedback (-.6) when entailed and implicated functions are included in the analysis. The weak negative association means that an utterance does not often have communicative functions in these two dimensions simultaneously. Some negative associations become positive if we take entailed and implicated functions into account, because, as already noted, dialogue acts like answers, accepts and rejects, imply positive feedback.

Dimensions	Task	Auto-F.	Allo-F.	Turn M.	Time M.	Contact M.	DS	OCM	PCM	SOM
Task		.1	.3	.06	-.4	-.6	.03	-.03	-.1	.04
Auto-F.	-.5		-.6	.1	-.3	.2	-.02	-.02	-.1	.04
Allo-F.	-.2	-.03		.09	-.1	-.2	.03	-.01	-.02	-.01
Turn M.	-.03	-.04	.14		.6	.04	-.06	.02	.02	-.03
Time M.	-.4	-.06	.14	.6		-.1	-.02	.04	-.03	-.02
Contact M.	-.05	-.006	-.003	.001	-.007		.04	-.01	-.04	-.03
DS	-.2	-.02	-.01	-.01	-.02	-.002		-.01	-.01	.2
OCM	.01	-.05	.02	.4	-.03	-.006	-.003		-.03	-.007
PCM	-.1	-.01	.01	-.006	.01	-.001	-.005	-.01		-.003
SOM	-.1	-.01	-.007	-.02	-.02	-.001	.05	-.007	-.003	

Table 2: *Extent of relation between dimensions for AMI corpus expressed in the Phi coefficient (implicated and entailed functions excluded (white cells) and included (grey cells)).*

The third independence test, mentioned above, shows that each dimension may be addressed by an utterance which does not address any other dimension. The Task dimension is independently addressed in 28.8% of the utterances; 14.2% of the utterances have a function in the Auto-Feedback dimension only; for the other dimensions these figures are 0.7% - Allo-Feedback; 7.4% - Turn Management; 0.3% - Time Management; 0.1% - Contact Management; 1.9% - Discourse Structuring; 0.5% - OCM; 0.2% - PCM; and 0.3% - SOM.

within	Task	Auto-F.	Allo-F.	Turn M.	Time M.	Contact M.	DS	OCM	PCM	SOM
Task	21.2	27.4	27.7	20	32.5	0	7.1	16.4	15.2	32.1
Auto-F.	15	24.4	25	21.4	15.4	27.8	12.3	7.5	22.7	12.8
Allo-F.	0.4	1.3	5.6	0.5	0.5	0	0.6	0.4	0	0
Turn M.	14.3	4.7	0	6.5	5.2	0	6.5	2.2	7.6	6.4
Time M.	22.2	16.3	16.7	23.5	15	0	35.5	47.1	37.9	19.2
Contact M.	0	0.1	0	0.2	0	27.8	0	0	0	0
DS	2	2	0	0.5	0.5	27.8	5.2	0.4	0	0
OCM	7.7	6.3	5.6	7.7	11.2	0	0	7	0	0
PCM	0.4	0.4	0	0	0.08	0	0	0.2	0	0
SOM	0.1	0.3	0	1.2	0.08	0	0.6	0	0	6.4

Table 3: *Overview of relative frequency (in%) of pairs of dimension and previous dimensions by previous utterances observed in AMI data, per dimension, drawn from the set of 5 pairs from the dialogue history.*

We finally investigated the occurrences of tags given the tags of the previous utterances, taking five previous utterances into account. Table 3 shows no evidence of dependences across the dialogue history. There are some frequent patterns, for example, retractions and self-corrections often follow hesitations because the speaker, while monitoring his own speech and noticing that part of it needs revision, needs time to construct the corrected part.

4 Conclusions

In this paper we investigated the independence of the dimensions defined in the DIT++ dialogue act annotation scheme, using co-occurrences matrices and the phi coefficient for measuring relatedness between dimensions.

The results show that, although some dimensions are more related and co-occur more frequently than others, on the whole the ten DIT++ dimensions may be considered to be independent aspects of communication.

Acknowledgments

This research was conducted as part of ISO project 24617-2: Semantic annotation framework, Part 2: Dialogue acts, and sponsored by Tilburg University.

References

James F. Allen and Mark G. Core. 1997. Draft of DAMSL: Dialog Act Markup in Several Layers.

Jens Allwood. 2000. An activity-based approach to pragmatics. In Bunt, H., and Black, W. (eds.) *Abduction, Belief and Context in Dialogue; Studies in Computational Pragmatics*, pp. 47–80. Benjamins, Amsterdam.

Harry Bunt and Yann Girard. 2005. Designing an open, multidimensional dialogue act taxonomy. In *Gardent, C., and Gaiffe, B. (eds). Proc. 9th Workshop on the Semantics and Pragmatics of Dialogue*, pp. 37–44.

Harry Bunt. 2006. Dimensions in dialogue annotation. In *Proceedings of LREC 2006*.

Mark G. Core and James F. Allen. 1997. Coding dialogues with the DAMSL annotation scheme. In *Working Notes: AAAI Fall Symposium on Communicative Action in Humans and Machines*, pp. 28–35.

Staffan Larsson. 1998. Coding Schemas for Dialogue Moves. *Technical report from the S-DIME project*.

Volha Petukhova and Harry Bunt. 2009. Dimensions in communication. *TiCC Technical Report 2009-002*, Tilburg University.

Andrei Popescu-Belis. 2005. Dialogue Acts: One or More Dimensions? *ISSCO Working Paper 62, ISSCO*.

Improving Coreference Resolution by Using Conversational Metadata

Xiaoqiang Luo and Radu Florian and Todd Ward
IBM T.J. Watson Research Center
Yorktown Heights, NY 10598
{xiaoluo,raduf,toddward}@us.ibm.com

Abstract

In this paper, we propose the use of metadata contained in documents to improve coreference resolution. Specifically, we quantify the impact of speaker and turn information on the performance of our coreference system, and show that the metadata can be effectively encoded as features of a statistical resolution system, which leads to a statistically significant improvement in performance.

1 Introduction

Coreference resolution aims to find the set of linguistic expressions that refer to a common entity. It is a discourse-level task given that the ambiguity of many referential relationships among linguistic expressions can only be correctly resolved by examining information extracted from the entire document.

In this paper, we focus on exploiting the structural information (e.g., speaker and turn in conversational documents) represented in the metadata of an input document. Such metadata often coincides with the discourse structure, and is presumably useful to coreference resolution. The goal of this study is to quantify the effect metadata. To this end, information contained in metadata is encoded as features in our coreference resolution system, and statistically significant improvement is observed.

The rest of the paper is organized as follows. In Section 2 we describe the data set on which this study is based. In Section 3 we first show how to incorporate information carried by metadata into a statistical coreference resolution system. We also quantify the impact of metadata when they are treated as extraneous data. Results and discussions of the results are also presented in that section.

2 Data Set

This study uses the 2007 ACE data. In the ACE program, a `mention` is textual reference to an object of interest while the set of mentions in a document referring to the same object is called `entity`. Each mention is of one of 7 entity types: FAC(cility), GPE (Geo-Political Entity), LOC(ation), ORG(anization), PER(son), VEH(icle), and WEA(pon). Every entity type has a predefined set of subtypes. For example, ORG subtypes include `commercial`, `governmental` and `educational` etc, which reflect different subgroups of organizations. Mentions referring to the same entity share the same type and subtype. A mention can also be assigned with one of 3 mention types: either `NAM`(e), `NOM`(inal), or `PRO`(noun). Accordingly, entities have "levels:" if an entity contains at least one `NAM` mention, its level is `NAM`; or if it does not contain any `NAM` mention, but contains at least one `NOM` mention, then the entity is of level `NOM`; if an entity has only `PRO` mention(s), then its level is `PRO`. More information about ACE entity annotation can be found in the official annotation guideline (Linguistic Data Consortium, 2008).

The ACE 2007 documents come from a variety of sources, namely newswire, broadcast conversation, broadcast news, Usenet, web log and telephone conversation. Some of them contain rich metadata, as illustrated in the following excerpt of one broadcast conversation document:

```
<DOC>
<DOCID>CNN_CF_20030303.1900.00</DOCID>
<TEXT>
<TURN>
<SPEAKER> Begala </SPEAKER>
Well, we'll debate that later on in the
show. We'll have a couple of experts
come out, ...
```


Proceedings of NAACL HLT 2009: Short Papers, pages 201–204,
Boulder, Colorado, June 2009.

```
</TURN>
<TURN>
<SPEAKER> Novak </SPEAKER>
Paul, as I understand your definition
of a political -- of a professional
politician based on that is somebody
who is elected to public office. ...
</TURN>
...
</TEXT>
</DOC>
```

In this example, SPEAKER and TURN information are marked by their corresponding SGML tags. Such metadata provides structural information: for instance, the metadata implies that Begala is the speaker of the utterance "Well, we'll debate ..., " and Novak the speaker of the utterance "Paul, as I understand your definition ..." Intuitively, knowing the speakers of the previous and current turn would make it a lot easier to find the right antecedent of pronominal mentions I and your in the sentence: "Paul, as I understand your definition ..."

Documents in non-conversational genres (e.g. newswire documents) also contain speaker and quotation, which resemble conversational utterance, but they are not annotated. For these documents, we use heuristics (e.g., existence of double or single quote, a short list of communication verb lemmas such as "say," "tell" and "speak" etc) to determine the speaker of a direct quotation if necessary.

3 Impact of Metadata

In this section we describe how metadata is used to improve our statistical coreference resolution system.

3.1 Resolution System

The coreference system used in our study is a data-driven, machine-learning-based system. Mentions in a document are processed sequentially by mention type: NAM mentions are processed first, followed by NOM mentions and then PRO mentions. The first mention is used to create an initial entity with a deterministic score 1. The second mention can be either linked to the first entity, or used to create a new entity, and the two actions are assigned a score computed from a log linear model. This process is repeated until all mentions in a document are processed. During training time, the process is applied to the training data and training instances (both positive and negative) are generated. At testing time, the same process is applied to an input document and the hypothesis with the highest score is selected as the final coreference result. At the core of the coreference system is a conditional log linear model $P(l|e, m)$ which measures how likely a mention m is or is not coreferential with an existing entity e. The modeling framework provides us with the flexibility to integrate metadata information by encoding it as features.

The coreference resolution system employs a variety of lexical, semantic, distance and syntactic features(Luo et al., 2004; Luo and Zitouni, 2005). The full-blown system achieves an 56.2% ACE-value score on the official 2007 ACE test data, which is about the same as the best-performing system in the Entity Detection and Recognition (EDR) task (NIST, 2007). So we believe that the resolution system is fairly solid.

The aforementioned 56.2% score includes mention detection (i.e., finding mention boundaries and predicting mention attributes) and coreference resolution. Since this study is about coreference resolution only, the subsequent experiments, are thus performed on gold-standard mentions. We split the ACE 2007 data into a training set consisting of 499 documents, and a test set of 100 documents. The training and test split ratio is roughly the same across genres. The performance numbers reported in the subsequent subsections are on the 100-document development test set.

3.2 Metadata Features

For conversational documents with speaker and turn information, we compute a group of binary features for a candidate referent r and the current mention m. Feature values are 1 if the conditions described below hold:

- if r is a speaker, m is a pronominal mention and r utters the sentence containing m.
- if r is a speaker, m is pronoun and r utters the sentence one turn before the one containing m.
- if mention r and mention m are seen in the same turn.
- if mention r and mention m are in two consecutive turns.

Note that the first feature is not subsumed by the third one since a turn may contain multiple sentences. For the same reason, the last feature does not subsume the second one. For the sample document in Section 2, the first feature fires if $r =$ Novak and $m =$ I; the second features fires if $r =$ Begala

and $m =$ I; the third feature fires if $r =$ Paul and $m =$ I; and lastly, the fourth feature fires if $r =$ We and $m =$ I. For ACE documents that do not carry turn and speaker information such as newswire, we use heuristic rules to empirically determine the speaker and the corresponding quotations before computing these features.

To test the effect of the feature group, we trained two models: a baseline system without speaker and turn features, and a contrast system by adding the speaker and turn features to the baseline system. The contrast results are tabulated in Table 1. We observe an overall 0.7 point ACE-value improvement. We also compute the ACE-values at document level for the two systems, and a paired Wilcoxon (Wilcoxon, 1945) rank-sum test is conducted, which indicates that the difference between the two systems is statistically significant at level $p \leq 0.002$.

Note that the features often help link pronouns with their antecedents in conversational documents. But ACE-value is a weighted metric which heavily discounts pronominal mentions and entities. We suspect that the effect of speaker and turn information could be larger if we weigh all mention types equally. This is confirmed when we looked at the unweighted B^3 (Bagga and Baldwin, 1998) numbers reported by the official ACE08 scorer (column B^3 in Table 1): the overall B^3 score is improved from 73.8% to 76.4% – a 2.6 point improvement, which is almost 4 times as large as the ACE-value change.

System	ACE-Value	B^3
baseline	78.7	73.8
+ Spkr/Turn	79.4	76.4

Table 1: Coreference performance: baseline vs. system with speaker and turn features.

3.3 Metadata: To Use Or Not to Use?

In the ACE evaluations prior to 2008, mentions inside metadata (such as speaker and poster) are annotated and scored as normal mentions, although such metadata is not part of the actual content of a document. An interesting question is: how large an effect do mentions inside metadata have on the system performance? If metadata are not annotated as mentions, is it still useful to look into them? To answer this question, we remove speaker mentions in conversational documents (i.e., broadcast conversation and telephone conversation) from both the training and test data. Then we train two systems:

- System A: the system totally disregards metadata.
- System B: the system first recovers speaker metadata using a very simple rule: all tokens within the <SPEAKER> tags are treated as one PER mention. This rule recovers most speaker mentions, but it can occasionally result in errors. For instance, the speaker "CNN correspondent John Smith" includes affiliation and profession information and ought to be tagged as three mentions: "CNN" as an ORG(anization) mention, "correspondent" and "John Smith" as two PER mentions. With recovered speaker mentions, we train a model and resolve coreference as normal.

After mentions in the test data are chained in System B, speaker mentions are then removed from system output so that the coreference result is directly comparable with that of System A.

The ACE-value comparison between System A and System B is shown in Table 2. As can be seen, System B works much better than System A, which ignores SPEAKER tags. For telephone conversations (cts), ACE-value improves as much as 4.6 points. A paired Wilcoxon test on document-level ACE-values indicates that the difference is statistically significant at $p < 0.016$.

System	bc	cts
A	75.2	66.8
B	76.6	71.4
Abs. Change	1.4	4.6

Table 2: Metadata improves the ACE-value for broadcast conversation (bc) and telephone conversation (cts) documents.

The reason why metadata helps is that speaker mention can be used to localize the coreference process and therefore improves the performance. For example, in the sentences uttered by "Novak" (cf. the sample document in Section 2), it is intuitively straightforward to link mention I with Novak, and your with Begala – when speaker mentions are made present in the coreference system B. On the other hand, in System A, "I" is likely to be linked with "Paul" because of its proximity of "Paul" in the absence of speaker information.

The result of this experiment suggests that, unsurprisingly, speaker and turn metadata carry structural

information helpful for coreference resolution. Even if speaker mentions are not annotated (as in System A), it is still beneficial to make use of it, e.g., by first identifying them automatically as in System B.

4 Related Work

There is a large body of literature for coreference resolution based on machine learning (Kehler, 1997; Soon et al., 2001; Ng and Cardie, 2002; Yang et al., 2008; Luo et al., 2004) approach. Strube and Muller (2003) presented a machine-learning based pronoun resolution system for spoken dialogue (Switchboard corpus). The document genre in their study is similar to the ACE telephony conversation documents, and they did include some dialogue-specific features, such as an anaphora's preference for S, VP or NP, in their system, but they did not use speaker or turn information. Gupta et al. (2007) presents an algorithm disambiguating generic and referential "you."

Cristea et al. (1999) attempted to improve coreference resolution by first analyzing the discourse structure of a document with rhetoric structure theory (RST) (Mann and Thompson, 1987) and then using the resulted discourse structure in coreference resolution. Since obtaining reliably the discourse structure itself is a challenge, they got mixed results compared with a linear structure baseline.

Our work presented in this paper concentrates on the structural information represented in metadata, such as turn or speaker information. Such metadata provides reliable discourse structure, especially for conversational documents, which is proven beneficial for enhancing the performance of our coreference resolution system.

Acknowledgments

This work is partially supported by DARPA GALE program under the contract number HR0011-06-02-0001. We'd also like to thank 3 reviewers for their helpful comments.

References

Amit Bagga and Breck Baldwin. 1998. Algorithms for scoring coreference chains. In *Proceedings of the Linguistic Coreference Workshop at The First International Conference on Language Resources and Evaluation (LREC'98)*, pages 563–566.

Dan Cristea, Nancy lde, Daniel Marcu, Valentin Tablanlivia Polanyi, and Martin van den Berg. 1999. Discourse structure and co-reference: An empirical study. In *Proceedings of ACL Workshop "The Relation of Discourse/Dialogue Structure and Reference"*. Association for Computational Linguistics.

Surabhi Gupta, Matthew Purver, and Dan Jurafsky. 2007. Disambiguating between generic and referential "you" in dialog. In *Proceedings of the 45th ACL(the Demo and Poster Sessions)*, pages 105–108, Prague, Czech Republic, June. Association for Computational Linguistics.

Andrew Kehler. 1997. Probabilistic coreference in information extraction. In *Proc. of EMNLP*.

Linguistic Data Consortium. 2008. ACE (Automatic Content Extraction) English annotation guidelines for entities. http://projects.ldc.upenn.edu/ace/docs/English-Entities-Guidelines_v6.5.pdf.

Xiaoqiang Luo and Imed Zitouni. 2005. Multilingual coreference resolution with syntactic features. In *Proc. of Human Language Technology (HLT)/Empirical Methods in Natural Language Processing (EMNLP)*.

Xiaoqiang Luo, Abe Ittycheriah, Hongyan Jing, Nanda Kambhatla, and Salim Roukos. 2004. A mention-synchronous coreference resolution algorithm based on the bell tree. In *Proc. of ACL*.

William C. Mann and Sandra A. Thompson. 1987. Rhetorical structure theory: A theory of text organization. Technical Report RS-87-190, USC/Information Sciences Institute.

Vincent Ng and Claire Cardie. 2002. Improving machine learning approaches to coreference resolution. In *Proc. of ACL*, pages 104–111.

NIST. 2007. 2007 automatic content extraction evaluation official results. http://www.nist.gov/speech/tests/ace/2007/doc/ace07_eval_official_results_20070402.html.

Wee Meng Soon, Hwee Tou Ng, and Chung Yong Lim. 2001. A machine learning approach to coreference resolution of noun phrases. *Computational Linguistics*, 27(4):521–544.

Michael Strube and Christoph Muller. 2003. A machine learning approach to pronoun resolution in spoken dialogue. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*.

Frank Wilcoxon. 1945. Individual comparisons by ranking methods. *Biometrics*, I:80–83.

Xiaofeng Yang, Jian Su, Jun Lang, Chew Lim Tan, Ting Liu, and Sheng Li. 2008. An entity-mention model for coreference resolution with inductive logic programming. In *Proceedings of ACL-08: HLT*, pages 843–851, Columbus, Ohio, June. Association for Computational Linguistics.

Using N-gram based Features for Machine Translation System Combination

Yong Zhao[1]
Georgia Institute of Technology
Atlanta, GA 30332, USA
yongzhao@gatech.edu

Xiaodong He
Microsoft Research
Redmond, WA 98052, USA
xiaohe@microsoft.com

Abstract

Conventional confusion network based system combination for machine translation (MT) heavily relies on features that are based on the measure of agreement of words in different translation hypotheses. This paper presents two new features that consider agreement of n-grams in different hypotheses to improve the performance of system combination. The first one is based on a sentence specific online n-gram language model, and the second one is based on n-gram voting. Experiments on a large scale Chinese-to-English MT task show that both features yield significant improvements on the translation performance, and a combination of them produces even better translation results.

1 Introduction

In past years, the confusion network based system combination approach has been shown with substantial improvements in various machine translation (MT) tasks (Bangalore, et. al., 2001, Matusov, et. al., 2006, Rosti, et. al., 2007, He, et. al., 2008). Given hypotheses of multiple systems, a confusion network is built by aligning all these hypotheses. The resulting network comprises a sequence of correspondence sets, each of which contains the alternative words that are aligned with each other. To derive a consensus hypothesis from the confusion network, decoding is performed by selecting a path with the maximum overall confidence score among all paths that pass the confusion network (Goel, et. al., 2004).

[1] The work was performed when Yong Zhao was an intern at Microsoft Research

The confidence score of a hypothesis could be assigned in various ways. Fiscus (1997) used voting by frequency of word occurrences. Mangu et. al., (2000) computed a word posterior probability based on voting of that word in different hypotheses. Moreover, the overall confidence score is usually formulated as a log-linear model including extra features including language model (LM) score, word count, etc.

Features based on word agreement measure are extensively studied in past work (Matusov, et. al., 2006, Rosti, et. al., 2007, He, et. al., 2008). However, utilization of n-gram agreement information among the hypotheses has not been fully explored yet. Moreover, it was argued that the confusion network decoding may introduce undesirable spur words that break coherent phrases (Sim, et. al., 2007). Therefore, we would prefer the consensus translation that has better n-gram agreement among outputs of single systems.

In the literature, Zens and Ney (2004) proposed an n-gram posterior probability based LM for MT. For each source sentence, a LM is trained on the n-best list produced by a single MT system and is used to re-rank that n-best list itself. On the other hand, Matusov et al. (2008) proposed an "adapted" LM for system combination, where this "adapted" LM is trained on translation hypotheses of the whole test corpus from all single MT systems involved in system combination.

Inspired by these ideas, we propose two new features based on n-gram agreement measure to improve the performance of system combination. The first one is a sentence specific LM built on translation hypotheses of multiple systems; the second one is n-gram-voting-based confidence. Experimental results are presented in the context of a large-scale Chinese-English translation task.

Proceedings of NAACL HLT 2009: Short Papers, pages 205–208,
Boulder, Colorado, June 2009.

2 System Combination for MT

One of the most successful approaches for system combination for MT is based on confusion network decoding as described in (Rosti, et. al., 2007). Given translation hypotheses from multiple MT systems, one of the hypotheses is selected as the backbone for the use of hypothesis alignment. This is usually done by a sentence-level minimum Bayes risk (MBR) re-ranking method. The confusion network is constructed by aligning all these hypotheses against the backbone. Words that align to each other are grouped into a correspondence set, constituting competition links of the confusion network. Each path in the network passes exactly one link from each correspondence set. The final consensus output relies on a decoding procedure that chooses a path with the maximum confidence score among all paths that pass the confusion network.

The confidence score of a hypothesis is usually formalized as a log-linear sum of several feature functions. Given a source language sentence F, the total confidence of a target language hypothesis $E = (e_1, \dots, e_L)$ in the confusion network can be represented as:

$$\begin{aligned} logP(E|F) = & \sum_{l=1}^{L} log\, P(e_l|l,F) \\ & + \lambda_1 logP_{LM}(E) \\ & + \lambda_2 N_{words}(E) \end{aligned} \quad (1)$$

where the feature functions include word posterior probability $P(e_l|l,F)$, LM probability $P_{LM}(E)$, and the number of real words N_{words} in E. Usually, the model parameter λ_i could be trained by optimizing an evaluation metric, e.g., BLEU score, on a held-out development set.

3 N-gram Online Language Model

Given a source sentence F, the fractional count $C(e_1^n|F)$ of an n-gram e_1^n is defined as:

$$C(e_1^n|F) = \sum_{E' \in \boldsymbol{E}^h} \sum_{l=n}^{L} P(E'|F)\, \delta(e'^{l}_{l-n+1}, e_1^n) \quad (2)$$

where $\boldsymbol{E}^h$ denotes the hypothesis set, $\delta(\cdot,\cdot)$ denotes the Kronecker function, and $P(E'|F)$ is the posterior probability of translation hypothesis E', which is expressed as the weighted sum of the system specific posterior probabilities through the systems that contains hypothesis E',

$$P(E|F) = \sum_{k=1}^{K} w_k P(E|S_k, F) 1(E \in \boldsymbol{E}_{S_k}) \quad (3)$$

where w_k is the weight for the posterior probability of the k^{th} system S_k, and $1(\cdot)$ is the indicator function.

Follows Rosti, et. al. (2007), system specific posteriors are derived based on a rank-based scoring scheme. I.e., if translation hypothesis E_r is the r^{th} best output in the n-best list of system S_k, posterior $P(E_r|S_k,F)$ is approximated as:

$$P(E_r|S_k,F) = \frac{1/(1+r)^{\eta}}{\sum_{r'=1}^{||\boldsymbol{E}_{S_k}||} 1/(1+r')^{\eta}} \quad (4)$$

where η is a rank smoothing parameter.

Similar to (Zens and Ney, 2004), a straightforward approach of using n-gram fractional counts is to formulate it as a sentence specific online LM. Then the online LM score of a path in the confusion network will be added as an additional feature in the log-linear model for decoding. The online n-gram LM score is computed by:

$$P(e_l|e_{l-n+1}^{l-1}, F) = \frac{C(e_{l-n+1}^{l}|F)}{C(e_{l-n+1}^{l-1}|F)} \quad (5)$$

The LM score of hypothesis E is obtained by:

$$P_{LM}(E|F) = \prod_{l=n}^{L} P\left(e_l|e_{l-n+1}^{l-1}, F\right) \quad (6)$$

Since new n-grams unseen in original translation hypotheses may be proposed by the CN decoder, LM smoothing is critical. In our approach, the score of the online LM is smoothed by taking a linear interpolation to combine scores of different orders.

$$P_{smoot\,h}\left(e_l|e_{l-n+1}^{l-1},F\right) = \sum_{m=1}^{n}\alpha_m P(e_l|e_{l-m+1}^{l-1},F) \quad (7)$$

In our implementation, the interpolation weights $\{\alpha_m\}$ can be learned along with other combination parameters in the same Max-BLEU training scheme via Powell's search.

4 N-gram-Voting-Based Confidence

Motivated by features based on voting of single word, we proposed new features based on N-gram voting. The voting score $V(e_1^n|F)$ of an n-gram e_1^n is computed as:

$$V(e_1^n|F) = \sum_{E'\in \boldsymbol{E}^h} P(E'|F)1(e_1^n \in E') \quad (8)$$

It receives a vote from each hypothesis that contains that n-gram, and weighted by the posterior probability of that hypothesis, where the posterior probability $P(E'|F)$ is computed by (3). Unlike the fractional count, each hypothesis can vote no more than once on an n-gram.

$V(e_1^n|F)$ takes a value between 0 and 1. It can be viewed as the confidence of the n-gram e_1^n. Then the n-gram-voting-based confidence score of a hypothesis E is computed as the product of confidence scores of n-grams in E:

$$P_{NV,n}(E|F) = P_{NV,n}\left(e_1^l|l,F\right) = \prod_{m=1}^{l-n+1} V(e_m^{m+n-1}|F) \quad (9)$$

where n can take the value of 2, 3, …, N. In order to prevent zero confidence, a small back-off confidence score is assigned to all n-grams unseen in original hypotheses.

Augmented with the proposed n-gram based features, the final log-linear model becomes:

$$\begin{aligned} &logP(E|F) \\ &= \sum_{l=1}^{L} log\,P(e_l|l,F) + \lambda_1 logP_{LM}(E) \\ &+ \lambda_2 N_{words}(E) + \lambda_3 logP_{LM}(E|F) \\ &+ \sum_{n=2}^{N} \lambda_{n+2} logP_{NV,n}(E|F) \end{aligned} \quad (10)$$

5 Evaluation

We evaluate the proposed n-gram based features on the Chinese-to-English (C2E) test in the past NIST Open MT Evaluations. The experimental results are reported in case sensitive BLEU score (Papineni, et. al., 2002).

The dev set, which is used for system combination parameter training, is the newswire and newsgroup parts of NIST MT06, which contains a total of 1099 sentences. The test set is the "*current*" test set of NIST MT08, which contains 1357 sentences of newswire and web-blog data. Both dev and test sets have four reference translations per sentence.

Outputs from a total of eight single MT systems were combined for consensus translations. These selected systems are based on various translation paradigms, such as phrasal, hierarchical, and syntax-based systems. Each system produces 10-best hypotheses per translation. The BLEU score range for the eight individual systems are from 26.11% to 31.09% on the dev set and from 20.42% to 26.24% on the test set. In our experiments, a state-of-the-art system combination method proposed by He, et. al. (2008) is implemented as the baseline. The true-casing model proposed by Toutanova et al. (2008) is used.

Table 1 shows results of adding the online LM feature. Different LM orders up to four are tested. Results show that using a 2-gram online LM yields a half BLEU point gain over the baseline. However, the gain is saturated after a LM order of three, and fluctuates after that.

Table 2 shows the performance of using n-gram-voting-based confidence features. The best result of 31.01% is achieved when up to 4-gram confidence features are used. The BLEU score keeps improving when longer n-gram confidence features are added. This indicates that the n-gram voting based confidence feature is robust to high order n-grams.

We further experimented with incorporating both features in the log-linear model and reported the results in Table 3. Given the observation that the n-gram voting based confidence feature is more robust to high order n-grams, we further tested using different n-gram orders for them. As shown in Table 3, using 3-gram online LM plus 2~4-gram voting

based confidence scores yields the best BLEU scores on both dev and test sets, which are 37.98% and 31.35%, respectively. This is a 0.84 BLEU point gain over the baseline on the MT08 test set.

Table 1: Results of adding the n-gram online LM.

BLEU %	Dev	Test
Baseline	37.34	30.51
1-gram online LM	37.34	30.51
2-gram online LM	37.86	31.02
3-gram online LM	37.87	**31.08**
4-gram online LM	37.86	31.01

Table 2: Results of adding n-gram voting based confidence features.

BLEU %	Dev	Test
Baseline	37.34	30.51
+ 2-gram voting	37.58	30.88
+ 2~3-gram voting	37.66	30.96
+ 2~4-gram voting	37.77	**31.01**

Table 3: Results of using both n-gram online LM and n-gram voting based confidence features

BLEU %	Dev	Test
Baseline	37.34	30.51
2-gram LM + 2-gram voting	37.78	30.98
3-gram LM + 2~3-gram voting	37.89	31.21
4-gram LM + 2~4-gram voting	37.93	31.08
3-gram LM + 2~4-gram voting	37.98	**31.35**

6 Conclusion

This work explored utilization of n-gram agreement information among translation outputs of multiple MT systems to improve the performance of system combination. This is an extension of an earlier idea presented at the NIPS 2008 Workshop on Speech and Language (Yong and He 2008). Two kinds of n-gram based features were proposed. The first is based on an online LM using n-gram fractional counts, and the second is a confidence feature based on n-gram voting scores. Our experiments on the NIST MT08 Chinese-English task showed that both methods yield nice improvements on the translation results, and incorporating both kinds of features produced the best translation result with a BLEU score of 31.35%, which is a 0.84% improvement.

References

J.G. Fiscus, 1997. A post-processing system to yield reduced word error rates: Recognizer Output Voting Error Reduction (ROVER), in *Proc. ASRU*.

S. Bangalore, G. Bordel, and G. Riccardi, 2001. Computing consensus translation from multiple machine translation systems, in *Proc. ASRU*.

E. Matusov, N. Ueffing, and H. Ney, 2006. Computing consensus translation from multiple machine translation systems using enhanced hypotheses alignment, in *Proc. EACL*.

A.-V.I. Rosti, S. Matsoukas, and R. Schwartz, 2007. Improved Word-Level System Combination for Machine Translation. In *Proc. ACL*.

X. He, M. Yang, J. Gao, P. Nguyen, and R. Moore, 2008. Indirect-HMM-based hypothesis alignment for combining outputs from machine translation systems, in *Proc. EMNLP*.

L. Mangu, 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):373-400.

R. Zens and H. Ney, 2004. N-Gram posterior probabilities for statistical machine translation, in *Proc. HLT-NAACL*.

K.C. Sim, W.J. Byrne, M.J.F. Gales, H. Sahbi and P.C. Woodland, 2007. Consensus network decoding for statistical machine translation system combination. in *Proc. ICASSP*.

V. Goel, S. Kumar, and W. Byrne, 2004. Segmental minimum Bayes-risk decoding for automatic speech recognition. *IEEE transactions on Speech and Audio Processing*, vol. 12, no. 3.

K. Papineni, S. Roukos, T. Ward, and W. Zhu, 2002. BLEU: a method for automatic evaluation of machine translation. in *Proc. ACL*.

K. Toutanova, H. Suzuki and A. Ruopp. 2008. Applying Morphology Generation Models to Machine Translation. In *Proc. of ACL*.

Yong Zhao and Xiaodong He. 2008. System Combination for Machine Translation Using N-Gram Posterior Probabilities. *NIPS 2008 WORKSHOP on Speech and Language: Learning-based Methods and Systems.* Dec. 2008

E. Matusov, G. Leusch, R. E. Banchs, N. Bertoldi, D. Dechelotte, M. Federico, M. Kolss, Y. 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, Sept. 2008.

Language Specific Issue and Feature Exploration in Chinese Event Extraction

Zheng Chen **Heng Ji**
Department of Computer Science
The Graduate Center Queens College and The Graduate Center
The City University of New York
365 Fifth Avenue, New York, NY 10016, USA
zchen1@gc.cuny.edu hengji@cs.qc.cuny.edu

Abstract

In this paper, we present a Chinese event extraction system. We point out a language specific issue in Chinese trigger labeling, and then commit to discussing the contributions of lexical, syntactic and semantic features applied in trigger labeling and argument labeling. As a result, we achieved competitive performance, specifically, F-measure of 59.9 in trigger labeling and F-measure of 43.8 in argument labeling.

1 Introduction

In this paper we address the event extraction task defined in Automatic Content Extraction (ACE)[1] program. The ACE program defines the following terminology for event extraction task:

- Trigger: the word that most clearly expresses an event's occurrence
- Argument: an *entity*, or a *temporal expression* or a *value* that plays a certain role in the event instance
- Event mention: a phrase or sentence with a distinguished trigger and participant arguments

Some English event extraction systems based on supervised learning have been reported by researchers (Ahn, 2006; Ji and Grishman, 2008). In this paper we developed a modularized Chinese event extraction system. We nicely handled the language specific issue in trigger labeling and explored effective lexical, syntactic and semantic features that were applied in trigger labeling and argument labeling. Tan et al. (2008) addressed the same task as we did in this paper. However, to our knowledge, the language specific issue and feature contributions for Chinese event extraction have not been reported by earlier researchers.

The remainder of the paper is organized as follows. Section 2 points out a language specific issue in Chinese trigger labeling and discusses two strategies of trigger labeling: word-based and character-based. Section 3 presents argument labeling. Section 4 discusses the experimental results. Section 5 concludes the paper.

2 Trigger Labeling

We split trigger labeling into two steps: 1) *trigger identification*: to recognize the event trigger 2) *trigger classification*: to assign an event type for the trigger. The two strategies we will discuss in trigger labeling (word-based and character-based) only differ in the first step.

2.1 A Language-Specific Issue

Chinese, and some other languages, e.g., Japanese do not have delimiters between words. Thus, segmentation is usually an indispensible step for further processing, e.g., Part-of-Speech tagging, parsing, etc. However, the segmentation may cause a problem in some tasks, e.g., name entity recognition (Jing et al., 2003) and event trigger identification. For a specific example, "*击毙*" (*shoot and kill*) is segmented as a Chinese word. However, there are two triggers in the word, one is "*击*"(*shoot*) with the event type of *Attack*, and the other is "*毙*"(*kill*) with the event type of *Die*. The trigger may also cross two or more words, e.g., the trigger is "*公开信*" (*public letter*) which crosses two words, "*公开*" (*public*) and "*信*"(*letter*).

In the ACE Chinese corpus, 2902 triggers exactly one-to-one match their corresponding words,

[1] http://www.nist.gov/speech/tests/ace/

Proceedings of NAACL HLT 2009: Short Papers, pages 209–212,
Boulder, Colorado, June 2009.

meanwhile, 431 triggers are inconsistent with the words (either within the word, or across words). The inconsistency rate is as high as 13%.

We then discuss two strategies of trigger labeling, one is word-based in which we use a global errata table to alleviate the inconsistency problem, and the other is character-based which solves the inconsistency problem.

2.2 Word-based Trigger Labeling

We apply Maximum-Entropy based classifiers for *trigger identification* and *trigger classification*. The two classifiers share the same set of features:

- **Lexical features**: word, POS of the word, previous word + word, word + next word, previous POS + POS, and POS + next POS.
- **Syntactic features**: 1) *depth*: the depth of the trigger in the parse tree 2) *path to root*: the path from the leaf node of the trigger to the root in the parse tree 3) *sub-categorization* : the phrase structure expanded by the father of the trigger 4) *phrase type*: the phrase type of the trigger
- **Semantic dictionaries**: 1) *predicate existence*: a boolean value indicating the existence of trigger in a predicate list which is produced from Chinese Propbank (Xue and Palmer, 2008) 2) *synonym entry*: the entry number of the trigger in a Chinese synonym dictionary
- **Nearest entity information**: 1) the entity type of the *syntactically* nearest entity to the trigger in the parse tree 2) the entity type of the *physically* nearest entity to the trigger in the sentence

To deal with the language-specific issue in trigger identification, we construct a global errata table to record the inconsistencies existing in the training set. In the test procedure, if the scanned word has an entry in the errata table, we select the possible triggers in the entry as candidate triggers.

2.3 Character-based Trigger Labeling

Although the error table significantly helps to reduce segmentation inconsistencies, it is not a perfect solution since it only recognizes the inconsistencies existing in the training data.

To take a further step we build a separate character-based *trigger identification* classifier for comparison. We use a **MEMM** (Maximum Entropy Markov Model) to label each character with a tag indicating whether it is out of the trigger (O), or is the beginning of the trigger (B) or is a part of the trigger except the beginning (I). Our MEMM classifier performs sequential classification by assigning each character one of the three tags. We then apply Viterbi algorithm to decode the tag sequence and identify the triggers in the sequence.

Features used in our MEMM classifier include: the character, previous character, next character, previous tag and word-based features that the character carries. We apply the same set of features for *trigger classification* as used in word-based trigger labeling.

3 Argument Labeling

We also split argument labeling into two steps: 1) *argument identification*: to recognize an entity or a temporal expression or a value as an argument 2) *role classification*: to assign a role to the argument. We apply Maximum-Entropy based classifiers for the two steps and they share the same set of features:

- **Basic features**: trigger, event subtype of the event mention, type of the ACE entity mention, head word of the entity mention, combined value of event subtype and head word, combined value of event subtype and entity subtype.
- **Neighbor words**: 1) left neighbor word of the entity, temporal expression, or value 2) right neighbor word of the entity, temporal expression, or value
- **Syntactic features**: 1) *sub-categorization*: the phrase structure expanding the parent of the trigger 2) *position*: the relative position of the entity regarding to the trigger (before or after) 3) *path*: the minimal path from the entity to the trigger 4) *distance*: the shortest length from the entity to the trigger in the parse tree

4 Experimental Results

4.1 Data and Scoring Metric

We used 2005 ACE training corpus for our experiments. The corpus contains 633 Chinese documents. In this paper we follow the setting of ACE diagnostic tasks and use the ground truth entities, times and values for our training and testing.

We randomly selected 558 documents as training set and 66 documents as test set. For the training set, we reserved 33 documents as development set.

We define the following standards to determine the *correctness* of an event mention:

- *A trigger is correctly labeled* if its event type and offsets exactly match a reference trigger.
- *An argument is correctly labeled* if its event type, offsets, and role match the reference argument mention.

4.2 Overall System Performance

Table 1 shows the overall Precision (P), Recall (R) and F-Measure (F) scores of our baseline system (word-based system with only lexical features in trigger labeling and basic features in argument labeling), word-based system with full integrated features and character-based system with full integrated features.

Comparing to the Chinese event extraction system reported by (Tan et al., 2008), our scores are much lower. However, we argue that we apply much more strict evaluation metrics.

4.3 Comparison between Word-based and Character-based Trigger Labeling

Table 1 lists the comparison results between character-based and word-based trigger labeling. It indicates that the character-based method outperforms the word-based method, mostly due to the better performance in the step of *trigger identification* (3.3% improvement in F-Measure) with precision as high as 82.4% (14.3% improvement), and a little loss in recall (2.1%).

4.4 Feature Contributions for Trigger Labeling

Table 2 presents the feature contributions for word-based trigger labeling, and we observe similar feature contributions for character-based since it only differs from word-based in *trigger identification* and works similarly in *trigger classification* (we omit the results here). Table 2 shows that maintaining an errata table is an effective strategy for word-based *trigger identification* and dictionary resources improve the performance.

It is worth noting that the performance drops when integrating the syntactic features. Our explanation might be that the trigger, unlike the predicate in the semantic role labeling task, can not only be a verb, but also can be a noun or other types. Thus the syntactic position for the trigger in the parse tree is much more flexible than the predicate in Semantic Role Labeling. For this reason, syntactic features are not so discriminative in trigger labeling. Furthermore, the syntactic features cannot discriminate the word senses of a candidate trigger. In the following example,

S1:运动员 正在 ***进入*** *球场 准备 即将 到来 的 球赛*
The players are ***entering*** *the stadium to prepare for the coming game.*
S2:很多 农产品 还 没有 ***进入*** *市场 就 腐烂。*
Many farm products have been rotted before ***entering*** *the market.*

The word "*进入*" (*entering*) indicates a "Transport" event in sentence 1 but not in sentence 2. The phrase structures around the word "*进入*" in both sentences are exactly the same (VP→VP-NP). However, if an entity of "PERSON" appears ahead of "*进入*", the word "*进入*" is much more likely to be a trigger. Hence the features of *nearby entity information* could be effective.

4.5 Feature Contributions for Argument Labeling

Table 3 shows feature contributions for argument labeling after word-based trigger labeling and we also observe the same feature contributions for argument labeling after character-based trigger labeling (results are omitted). It shows that the two *neighbor word* features are fairly effective. We observe that in some patterns of event description, the left word is informative to tell the followed entity mention is an argument. For example, "*被 [Entity]打死*"(*killed by [Entity]*) is a common pattern to describe an attack event, and the left neighbor word of the entity "*被*" (*by*) can strongly imply that the entity is an argument with a role of "Attacker". Meanwhile, the right word can help reduce the spurious arguments. For example, in the Chinese "*的*" (*of*) structure, the word "*的*" (*of*) strongly suggests that the entity on the left side of "的" is not an argument.

The sub-categorization feature contributes little since it is a feature shared by all the arguments in the parse tree. Table 3 also shows that *Path* and *Distance* are two effective features. It is obvious that in the parse tree, each argument attached to the trigger is in a certain syntactic configuration. For example, the path "NP ↑ VP ↓ VV" implies that it might be a Subject-Verb structure and thus the entity in NP is highly likely to be an argument of the trigger (VV). The *Position* feature is helpful to discriminate argument roles in syntactically identical structure, e.g., "Subject Verb Object" structure.

Performance / System	Trigger Identification			Trigger Labeling			Argument Identification			Argument Labeling		
	P	R	F	P	R	F	P	R	F	P	R	F
Baseline	61.0	50.0	54.9	58.7	48.2	52.9	49.5	38.2	43.1	44.6	34.4	38.9
Word-based	68.1	52.7	59.4	65.7	50.9	57.4	56.1	38.2	45.4	53.1	36.2	43.1
Character-based	82.4	50.6	62.7	78.8	48.3	59.9	64.4	36.4	46.5	60.6	34.3	43.8

Table 1. Overall system performance (%)

	Trigger Identification			Trigger Labeling		
	P	R	F	P	R	F
Lexical features : (1)	61.0	50.0	54.9	58.7	48.2	52.9
(1) + Errata table: (2)	64.0	52.0	57.4	61.3	49.8	54.9
(2) + Dictionaries: (3)	64.9	53.5	58.6	62.7	51.6	56.6
(3)+ Syntactic features: (4)	64.3	51.8	57.4	60.6	48.9	54.1
(3) + Entity information: (5)	68.1	52.7	59.4	65.7	50.9	57.4

Table 2. Feature contributions for word-based trigger labeling (%)

	Argument Identification			Argument Labeling		
	P	R	F	P	R	F
Basic feature set: (1)	40.5	32.8	36.2	37.7	30.5	33.7
(1)+Left word: (2)	45.2	35.4	39.7	41.6	32.5	36.5
(1)+Right word: (3)	47.7	35.6	40.8	44.1	32.9	37.7
Feature set 2: (2)+(3)	**49.0**	**35.7**	**41.3**	**46.1**	**33.6**	**38.9**
(1)+Sub-categorization: (4)	41.9	33.1	37.0	38.7	30.5	34.1
(1)+Path: (5)	46.6	36.2	40.7	43.4	33.7	38.0
(1)+Distance: (6)	49.5	37.0	42.3	45.0	33.6	38.5
(1)+Position:(7)	43.8	35.3	39.1	41.0	33.1	36.6
Feature set 3 (from 4 to 7)	**56.2**	**36.1**	**43.9**	**51.2**	**32.9**	**40.0**
Total	**56.1**	**38.2**	**45.4**	**53.1**	**36.2**	**43.1**

Table 3. Feature contributions for argument labeling after word-based trigger labeling (%)

5 Conclusions and Future Work

In this paper, we took a close look at language specific issue in Chinese event extraction and explored effective features for Chinese event extraction task. All our work contributes to setting up a high performance Chinese event extraction system.

For future work, we intend to explore an approach to conducting cross-lingual event extraction and investigate whether the cross-lingual inference can bootstrap either side when running two language event extraction systems in parallel.

Acknowledgments

This material is based upon work supported by the Defense Advanced Research Projects Agency under Contract No. HR0011-06-C-0023 via 27-001022, and the CUNY Research Enhancement Program and GRTI Program.

References

D. Ahn. 2006. The stages of event extraction. *Proc. COLING/ACL 2006 Workshop on Annotating and Reasoning about Time and Events*. Sydney, Australia.

H. Ji and R. Grishman. 2008. Refining Event Extraction Through Cross-document Inference. *Proc. ACL 2008*. Ohio, USA.

H. Jing, R. Florian, X. Luo, T. Zhang, and A. Ittycheriah. 2003. HowtogetaChineseName(Entity): Segmentation and combination issues. *Proc. EMNLP 2003*.

H. Tan; T. Zhao; J. Zheng. 2008. Identification of Chinese Event and Their Argument Roles. *Proc. of the 2008 IEEE 8th International Conference on Computer and Information Technology Workshops*.

N. Xue and M. Palmer. 2008. Adding Semantic Role to the Chinese Treebank. *Natural Language Engineering. Combridge University Press.*

Improving A Simple Bigram HMM Part-of-Speech Tagger by Latent Annotation and Self-Training

Zhongqiang Huang[†], **Vladimir Eidelman**[†], **Mary Harper**[†‡]
[†]Laboratory for Computational Linguistics and Information Processing
Institute for Advanced Computer Studies
University of Maryland, College Park
[‡]Human Language Technology Center of Excellence
Johns Hopkins University
{zqhuang,vlad,mharper}@umiacs.umd.edu

Abstract

In this paper, we describe and evaluate a bigram part-of-speech (POS) tagger that uses latent annotations and then investigate using additional genre-matched unlabeled data for self-training the tagger. The use of latent annotations substantially improves the performance of a baseline HMM bigram tagger, outperforming a trigram HMM tagger with sophisticated smoothing. The performance of the latent tagger is further enhanced by self-training with a large set of unlabeled data, even in situations where standard bigram or trigram taggers do not benefit from self-training when trained on greater amounts of labeled training data. Our best model obtains a state-of-the-art Chinese tagging accuracy of 94.78% when evaluated on a representative test set of the Penn Chinese Treebank 6.0.

1 Introduction

Part-of-speech (POS) tagging, the process of assigning every word in a sentence with a POS tag (e.g., NN (normal noun) or JJ (adjective)), is prerequisite for many advanced natural language processing tasks. Building upon the large body of research to improve tagging performance for various languages using various models (e.g., (Thede and Harper, 1999; Brants, 2000; Tseng et al., 2005b; Huang et al., 2007)) and the recent work on PCFG grammars with latent annotations (Matsuzaki et al., 2005; Petrov et al., 2006), we will investigate the use of fine-grained latent annotations for Chinese POS tagging. While state-of-the-art tagging systems have achieved accuracies above 97% in English, Chinese POS tagging (Tseng et al., 2005b; Huang et al., 2007) has proven to be more challenging, and it is the focus of this study.

The value of the latent variable approach for tagging is that it can learn more fine grained tags to better model the training data. Liang and Klein (2008) analyzed the errors of unsupervised learning using EM and found that both estimation and optimization errors decrease as the amount of unlabeled data increases. In our case, the learning of latent annotations through EM may also benefit from a large set of automatically labeled data to improve tagging performance. Semi-supervised, self-labeled data has been effectively used to train acoustic models for speech recognition (Ma and Schwartz, 2008); however, early investigations of self-training on POS tagging have mixed outcomes. Clark et al. (2003) reported positive results with little labeled training data but negative results when the amount of labeled training data increases. Wang et al. (2007) reported that self-training improves a trigram tagger's accuracy, but this tagger was trained with only a small amount of in-domain labeled data.

In this paper, we will investigate whether the performance of a simple bigram HMM tagger can be improved by introducing latent annotations and whether self-training can further improve its performance. To the best of our knowledge, this is the first attempt to use latent annotations with self-training to enhance the performance of a POS tagger.

2 Model

POS tagging using a hidden Markov model can be considered as an instance of Bayesian inference,

Proceedings of NAACL HLT 2009: Short Papers, pages 213–216,
Boulder, Colorado, June 2009.

wherein we observe a sequence of words and need to assign them the most likely sequence of POS tags. If t_1^i denotes the tag sequence $t_1, \cdots, t_i$, and w_1^i denotes the word sequence $w_1, \cdots, w_i$, given the first-order Markov assumption of a bigram tagger, the best tag sequence $\tau(w_1^n)$ for sentence w_1^n can be computed efficiently as[1]:

$$\begin{aligned}\tau(w_1^n) &= \arg\max_{t_1^n} p(t_1^n|w_1^n) \\ &\approx \arg\max_{t_1^n} \prod_i p(t_i|t_{i-1})p(w_i|t_i)\end{aligned}$$

with a set of transition parameters $\{p(b|a)\}$, for transiting to tag b from tag a, and a set of emission parameters $\{p(w|a)\}$, for generating word w from tag a. A simple HMM tagger is trained by pulling counts from labeled data and normalizing to get the conditional probabilities.

It is well know that the independence assumption of a bigram tagger is too strong in many cases. A common practice for weakening the independence assumption is to use a second-order Markov assumption, i.e., a trigram tagger. This is similar to explicitly annotating each POS tag with the preceding tag. Rather than explicit annotation, we could use latent annotations to split the POS tags, similarly to the introduction of latent annotations to PCFG grammars (Matsuzaki et al., 2005; Petrov et al., 2006). For example, the NR tag may be split into NR-1 and NR-2, and correspondingly the POS tag sequence of "Mr./NR Smith/NR saw/VV Ms./NR Smith/NR" could be refined as: "Mr./NR-2 Smith/NR-1 saw/VV-2 Ms./NR-2 Smith/NR-1".

The objective of training a bigram tagger with latent annotations is to find the transition and emission probabilities associated with the latent tags such that the likelihood of the training data is maximized. Unlike training a standard bigram tagger where the POS tags are observed, in the latent case, the latent tags are not observable, and so a variant of EM algorithm is used to estimate the parameters.

Given a sentence w_1^n and its tag sequence t_1^n, consider the i-th word w_i and its latent tag $a_x \in a = t_i$ (which means a_x is a latent tag of tag a, the i-th tag in the sequence) and the $(i+1)$-th word w_{i+1} and its latent tag $b_y \in b = t_{i+1}$, the forward, $\alpha_{i+1}(b_y) = p(w_1^{i+1}, b_y)$, and backward, $\beta_i(a_x) = p(w_{i+1}^n|a_x)$, probabilities can be computed recursively:

$$\alpha_{i+1}(b_y) = \sum_x \alpha_i(a_x)p(b_y|a_x)p(w_{i+1}|b_y)$$

$$\beta_i(a_x) = \sum_y p(b_y|a_x)p(w_{i+1}|b_y)\beta_{j+1}(b_y)$$

In the E step, the posterior probabilities of co-occurrence events can be computed as:

$$\begin{aligned}p(a_x, b_y|w) &\propto \alpha_i(a_x)p(b_y|a_x)\beta_{i+1}(b_y) \\ p(a_x, w_i|w) &\propto \alpha_i(a_x)\beta_i(a_x)\end{aligned}$$

In the M step, the above posterior probabilities are used as weighted observations to update the transition and emission probabilities[2]:

$$\begin{aligned}p(b_y|a_x) &= c(a_x, b_y)/\sum_{b_y} c(a_x, b_y) \\ p(w|a_x) &= c(a_x, w)/\sum_w c(a_x, w)\end{aligned}$$

A hierarchical split-and-merge method, similar to (Petrov et al., 2006), is used to gradually increase the number of latent annotations while allocating them adaptively to places where they would produce the greatest increase in training likelihood (e.g., we observe heavy splitting in categories such as NN (normal noun) and VV (verb), that cover a wide variety of words, but only minimal splitting in categories like IJ (interjection) and ON (onomatopoeia)).

Whereas tag transition occurrences are frequent, allowing extensive optimization using EM, word-tag co-occurrences are sparser and more likely to suffer from over-fitting. To handle this problem, we map all words with frequency less than threshold[3] λ to symbol *unk* and for each latent tag accumulate the word tag statistics of these rare words to $c_r(a_x, \textit{unk}) = \sum_{w:c(w)<\lambda} c(a_x, w)$. These statistics are redistributed among the rare words $(w : c(w) < \lambda)$ to compute their emission probabilities:

$$\begin{aligned}c(a_x, w) &= c_r(a_x, \textit{unk}) \cdot c(a, w)/c_r(a, unk) \\ p(w|a_x) &= c(a_x, w)/\sum_w c(a_x, w)\end{aligned}$$

The impact of this rare word handling method will be investigated in Section 3.

A character-based unknown word model, similar to the one described in (Huang et al., 2007), is used to handle unknown Chinese words during tagging. A decoding method similar to the max-rule-product method in (Petrov and Klein, 2007) is used to tag sentences using our model.

3 Experiments

The Penn Chinese Treebank 6.0 (CTB6) (Xue et al., 2005) is used as the labeled data in our study. CTB6

[1] We assume that symbols exist implicitly for boundary conditions.

[2] $c(\cdot)$ represents the count of the event.

[3] The value of λ is tuned on the development set.

contains news articles, which are used as the primary source of labeled data in our experiments, as well as broadcast news transcriptions. Since the news articles were 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 them 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. The broadcast news data exhibits many of the characteristics of newswire text (it contains many nonverbal expressions, e.g., numbers and symbols, and is fully punctuated) and so is also included in the training data set. We also utilize a greater number of unlabeled sentences in the self-training experiments. They are selected from similar sources to the newswire articles, and are normalized (Zhang and Kahn, 2008) and word segmented (Tseng et al., 2005a). See Table 1 for a summary of the data used.

	Train	Dev	Test	Unlabeled
sentences	24,416	1904	1975	210,000
words	678,811	51,229	52,861	6,254,947

Table 1: The number of sentences and words in the data.

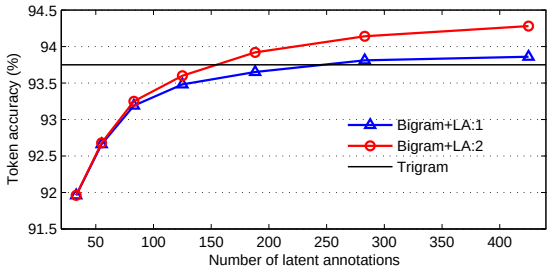


Figure 1: The learning curves of the bigram tagger with latent annotations on the development set.

Figure 1 plots the learning curves of two bigram taggers with latent annotations (Bigram+LA:2 has the special handling of rare words as described in Section 2 while Bigram+LA:1 does not) and compares its performance with a state-of-the-art trigram HMM tagger (Huang et al., 2007) that uses trigram transition and emission models together with bidirectional decoding. Both bigram taggers initially have much lower tagging accuracy than the trigram tagger, due to its strong but invalid independence assumption. As the number of latent annotations increases, the bigram taggers are able to learn more from the context based on the latent annotations, and their performance improves significantly, outperforming the trigram tagger. The performance gap between the two bigram taggers suggests that over-fitting occurs in the word emission model when more latent annotations are available for optimization; sharing the statistics among rare words alleviates some of the sparseness while supporting the modeling of deeper dependencies among more frequent events. In the later experiments, we use Bigram+LA to denote the Bigram+LA:2 tagger.

Figure 2 compares the self-training capability of three models (the bigram tagger w/ or w/o latent annotations, and the aforementioned trigram tagger) using different sizes of labeled training data and the full set of unlabeled data. For each model, a tagger is first trained on the allocated labeled training data and is then used to tag the unlabeled data. A new tagger is then trained on the combination[4] of the allocated labeled training data and the newly automatically labeled data.

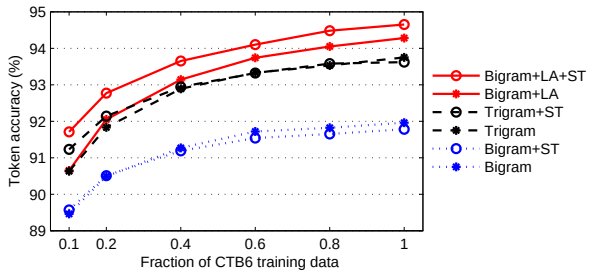


Figure 2: The performance of three taggers evaluated on the development set, before and after self-training with different sizes of labeled training data.

There are two interesting observations that distinguish the bigram tagger with latent annotations from the other two taggers. First, although all of the taggers improve as more labeled training data is available, the performance gap between the bigram tagger with latent annotations and the other two taggers also increases. This is because more latent annotations can be used to take advantage of the additional training data to learn deeper dependencies.

Second, the bigram tagger with latent annotations benefits much more from self-training, although it

[4]We always balance the size of manually and automatically labeled data through duplication (for the trigram tagger) or posterior weighting (for the bigram tagger w/ or w/o latent annotations), as this provides superior performance.

already has the highest performance among the three taggers before self-training. The bigram tagger without latent annotations benefits little from self-training. Except for a slight improvement when there is a small amount of labeled training, self-training slightly hurts tagging performance as the amount of labeled data increases. The trigram tagger benefits from self-training initially but eventually has a similar pattern to the bigram tagger when trained on the full labeled set. The performance of the latent bigram tagger improves consistently with self-training. Although the gain decreases for models trained on larger training sets, since stronger models are harder to improve, self-training still contributes significantly to model accuracy.

The final tagging performance on the test set is reported in Table 2. All of the improvements are statistically significant ($p < 0.005$).

Tagger	Token Accuracy (%)
Bigram	92.25
Trigram	93.99
Bigram+LA	94.53
Bigram+LA+ST	94.78

Table 2: The performance of the taggers on the test set.

It is worth mentioning that we initially added latent annotations to a trigram tagger, rather than a bigram tagger, to build from a stronger starting point; however, this did not work well. A trigram tagger requires sophisticated smoothing to handle data sparsity, and introducing latent annotations exacerbates the sparsity problem, especially for trigram word emissions. The uniform extension of a bigram tagger to a trigram tagger ignores whether the use of additional context is helpful and supported by enough data, nor is it able to use a longer context. In contrast, the bigram tagger with latent annotations is able to learn different granularities for tags based on the training data.

4 Conclusion

In this paper, we showed that the accuracy of a simple bigram HMM tagger can be substantially improved by introducing latent annotations together with proper handling of rare words. We also showed that this tagger is able to benefit from self-training, despite the fact that other models, such as bigram or trigram HMM taggers, do not.

In the future work, we will investigate automatic data selection methods to choose materials that are most suitable for self-training and evaluate the effect of the amount of automatically labeled data.

Acknowledgments

This work was supported by NSF IIS-0703859 and DARPA HR0011-06-C-0023 and HR0011-06-2-001. Any opinions, findings and/or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.

References

T. Brants. 2000. TnT a statistical part-of-speech tagger. In *ANLP*.

S. Clark, J. R. Curran, and M. Osborne. 2003. Bootstrapping pos taggers using unlabelled data. In *CoNLL*.

Z. Huang, M. Harper, and W. Wang. 2007. Mandarin part-of-speech tagging and discriminative reranking. *EMNLP*.

P. Liang and D. Klein. 2008. Analyzing the errors of unsupervised learning. In *ACL*.

J. Ma and R. Schwartz. 2008. Factors that affect unsupervised training of acoustic models. In *Interspeech*.

T. Matsuzaki, Y. Miyao, and J. Tsujii. 2005. Probabilistic CFG with latent annotations. In *ACL*. Association for Computational Linguistics.

S. Petrov and D. Klein. 2007. Improved inference for unlexicalized parsing. In *HLT-NAACL*.

S. Petrov, L. Barrett, R. Thibaux, and D. Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *ACL*.

S. M. Thede and M. P. Harper. 1999. A second-order hidden markov model for part-of-speech tagging. In *ACL*.

H. Tseng, P. Chang, G. Andrew, D. Jurafsky, and C. Manning. 2005a. A conditional random field word segmenter. In *SIGHAN Workshop on Chinese Language Processing*.

H. Tseng, D. Jurafsky, and C. Manning. 2005b. Morphological features help pos tagging of unknown words across language varieties. In *SIGHAN Workshop on Chinese Language Processing*.

W. Wang, Z. Huang, and M. Harper. 2007. Semi-supervised learning for part-of-speech tagging of Mandarin transcribed speech. In *ICASSP*.

N. Xue, F. Xia, F. Chiou, and M. Palmer. 2005. The penn chinese treebank: Phrase structure annotation of a large corpus. *Natural Language Engineering*.

B. Zhang and J. G. Kahn. 2008. Evaluation of decatur text normalizer for language model training. Technical report, University of Washington.

Statistical Post-Editing of a Rule-Based Machine Translation System*

A.-L. Lagarda, V. Alabau, F. Casacuberta
Instituto Tecnológico de Informática
Universidad Politécnica de Valencia, Spain
alagarda@iti.upv.es

R. Silva, and **E. Díaz-de-Liaño**
Celer Soluciones, S.L.
Madrid, Spain

Abstract

Automatic post-editing (*APE*) systems aim at correcting the output of machine translation systems to produce better quality translations, i.e. produce translations can be manually post-edited with an increase in productivity. In this work, we present an *APE* system that uses statistical models to enhance a commercial rule-based machine translation (*RBMT*) system. In addition, a procedure for effortless human evaluation has been established. We have tested the *APE* system with two corpora of different complexity. For the *Parliament* corpus, we show that the *APE* system significantly complements and improves the *RBMT* system. Results for the *Protocols* corpus, although less conclusive, are promising as well. Finally, several possible sources of errors have been identified which will help develop future system enhancements.

1 Introduction

Current machine translation systems are far from perfect. To achieve high-quality output, the raw translations they generate often need to be corrected, or post-edited by human translators. One way of increasing the productivity of the whole process is the development of automatic post-editing (*APE*) systems (Dugast et al., 2007; Simard et al., 2007).

* Work supported by the EC (FEDER) and the Spanish MEC under grant TIN2006-15694-CO2-01, by the Spanish research programme Consolider Ingenio 2010:MIPRCV (CSD2007-00018), and by the i3media Cenit project (CDTI 2007-1012).

Many of these works propose a combination of rule-based machine translation (*RBMT*) and statistical machine translation (*SMT*) systems, in order to take advantage of the particular capabilities of each system (Chen and Chen, 1997).

A possible combination is to automatically post-edit the output of a *RBMT* system employing a *SMT* system. In this work, we will apply this technique into two different corpora: *Parliament* and *Protocols*. In addition, we will propose a new human evaluation measure that will deal with the impact of the automatic post-editing.

This paper is structured as follows: after a brief introduction of the *RBMT*, *SMT*, and *APE* systems in Section 2, Section 3 details the carried out experimentation, discussing its results. Finally, some conclusions and future work are presented in Section 4.

2 Systems description

Three different systems are compared in this work, namely the *RBMT*, *SMT*, and *APE* systems.

Rule-based machine translation. *RBMT* was the first approach to machine translation, and thus, a relatively mature area in this field. *RBMT* systems are basically constituted by two components: the rules, that account for the syntactic knowledge, and the lexicon, which deals with the morphological, syntactic, and semantic information. Both rules and lexicons are grounded on linguistic knowledge and generated by expert linguists. As a result, the build process is expensive and the system is difficult to maintain (Bennett and Slocum, 1985). Furthermore, *RBMT* systems fail to adapt to new domains.

Proceedings of NAACL HLT 2009: Short Papers, pages 217–220,
Boulder, Colorado, June 2009.

Although they usually provide a mechanism to create new rules and extend and adapt the lexicon, changes are usually very costly and the results, frequently, do not pay off (Isabelle et al., 2007).

Statistical machine translation. In *SMT*, translations are generated on the basis of statistical models, which are derived from the analysis of bilingual text corpora. The translation problem can be statistically formulated as in (Brown et al., 1993). In practice, several models are often combined into a *log-linear* fashion. Each model can represent an important feature for the translation, such as *phrase-based*, *language*, or *lexical* models (Koehn et al., 2003).

Automatic post-editing. An *APE* system can be viewed as a translation process between the output from a previous MT system, and the target language. In our case, an *APE* system based on statistical models will be trained to correct the translation errors made by a *RBMT* system. As a result, both *RBMT* and *SMT* technologies will be combined in order to increase the overall translation quality.

3 Experiments

We present some experiments carried out using the introduced *APE* system, and comparing its performance with that of the *RBMT* and *SMT* systems. In the experimentation, two different English-to-Spanish corpora have been chosen, *Parliament* and *Protocols*, both of them provided by a professional translation agency.

Corpora. The *Parliament* corpus consists of a series of documents from proceedings of parliamentary sessions, provided by a client of the translation agency involved in this work. Most of the sentences are transcriptions of parliamentary speeches, and thus, with the peculiarities of the oral language. Despite of the multi-topic nature of the speeches, differences in training and test perplexities indicate that the topics in test are well represented in the training set (corpus statistics in Table 1).

On the other hand, the *Protocols* corpus is a collection of medical protocols. This is a more difficult task, as its statistics reflect in Table 1. There are many factors that explain this complexity, such as the different companies involved in training and test sets, out-of-domain test data (see perplexity and out-of-vocabulary words), non-native authors, etc.

Table 1: Corpus statistics for *Parliament* and *Protocols*. OOV stands for out-of-vocabulary words.

		Parliament		*Protocols*	
		En	Sp	En	Sp
Training	Sentences	90K	90K	154K	154K
	Run. words	2.3M	2.5M	3.2M	3.6M
	Vocabulary	29K	45K	41K	47K
	Perplexity	42	37	21	19
Test	Sentences	1K	1K	3K	3K
	Run. words	33K	33K	54K	71K
	OOVs	157	219	2K	1.7K
	Perplexity	44	43	131	173

Evaluation. In order to assess the proposed systems, a series of measures have been considered. In first place, some state-of-the-art automatic metrics have been chosen to give a first idea of the quality of the translations. These translations have been also evaluated by professional translators to assess the increase of productivity when using each system.

Automatic evaluation. The automatic assessment of the translation quality has been carried out using the *BiLingual Evaluation Understudy* (BLEU) (Papineni et al., 2002), and the *Translation Error Rate* (TER) (Snover et al., 2006). The latter takes into account the number of edits required to convert the system output into the reference. Hence, this measure roughly estimates the post-edition process.

Human evaluation. A new human evaluation measure has been proposed to roughly estimate the productivity increase when using each of the systems in a real scenario, grounded on previous works for human evaluation of qualitative factors (Callison-Burch et al., 2007). One of the desired qualities for this measure was that it should pose little effort to the human evaluator. Thus, a binary measure was chosen, the *suitability*, where the translations are identified as suitable or not suitable. A given translation is considered to be suitable if it can be manually post-edited with effort savings, i.e., the evaluator thinks that a manual post-editing will increase his productivity. On the contrary, if the evaluator prefers to ignore the proposed translation and start it over, the sentence is deemed not suitable.

Significance tests. Significance of the results has been assessed by the *paired bootstrap resampling* method, described in (Koehn, 2004). It estimates how confidently the conclusion that a system outperforms another one can be drawn from a test result.

Experimental setup. Rule-based translation was performed by means of a commercial *RBMT* system. On the other hand, statistical training and translation in both *SMT* and *APE* systems were carried out using the Moses toolkit (Koehn et al., 2007). It should be noted that *APE* system was trained taking the *RBMT* output as source, instead of the original text. In this way, it is able to post-edit the *RBMT* translations.

Finally, the texts employed for the human evaluation were composed by 350 sentences randomly drawn from each one of the two test corpora described in this paper. Two professional translators carried out the human evaluation.

3.1 Results and discussion

Experimentation results in terms of automatic and human evaluation are shown in this section.

Automatic evaluation. Table 2 presents *Parliament* and *Protocols* corpora translation results in terms of automatic metrics. Note that, as there is a single reference, this results are somehow pessimistic.

In the case of the *Parliament* corpus, *SMT* system outperforms the rest of the systems. *APE* results are slightly worse than *SMT*, but far better than *RBMT*.

However, when moving to the *Protocols* corpus, a more difficult task (as seen in perplexity in Table 1), the results show quite the contrary. *SMT* and *APE* systems show how they are more sensitive to out-of-domain documents. Nevertheless, the *RBMT* system seems to be more robust under such conditions. Despite of the degradation of the statistical models, *APE* manages to achieve much better results than the other two systems. It is able to conserve the robustness of *RBMT*, while its statistical counterpart deals with the particularities of the corpus.

Human evaluation. Table 3 shows the percentage of translations deemed suitable by the human evaluators. Two professional evaluators analysed the suitability of the output of each system

In the *Parliament* case, *APE* performance is found much more suitable than the rest of the systems. In fact, this difference between *APE* and the rest is statistically significant at a 99% level of confidence. In addition, significance tests show that, on average, *APE* improves *RBMT* on 59.5% of translations.

Table 2: Automatic evaluation for *Parliament* and *Protocols* tests.

	Parliament		*Protocols*	
	BLEU	TER	BLEU	TER
RBMT	29.1	46.7	29.5	48.0
SMT	49.9	34.9	22.4	59.6
APE	48.4	35.9	33.6	46.2

Regarding to the *Protocols* corpus, it must be noted that a first review of the translations pointed out that the *SMT* system performed quite poorly. Hence, *SMT* was not considered for the human evaluation on this corpus.

Figures show that *APE* complements and improves *RBMT*, although differences between them are tighter than in the *Parliament* corpus. However, significance tests still prove that these improvements are statistically significant (68% of confidence), and that the average improvement is 6.5%.

Table 3: Human evaluation for *Parliament* and *Protocols* corpora. Percentage of suitable translated sentences for each system.

	Parliament	*Protocols*
RBMT	58	60
SMT	60	–
APE	94	67

It is interesting to note how automatic measures and human evaluation seem not to be quite correlated. In terms of automatic measures, the best system to translate the *Parliament* test is the *SMT*. This improvement has been checked by carrying out significance tests, resulting statistically significant with a 99% of confidence. However, in the human evaluation, *SMT* is worse than *APE* (this difference is also significant at 99%). On the other hand, when working with the *Protocols* corpus, automatic metrics indicate that *APE* improves the rest (significant improvement at 99%). Nevertheless, human evaluators seem to think that the difference between *APE* and *RBMT* is not so significant, only with a confidence of 68%. Previous works confirm this apparent

discrepancy between automatic and human evaluations (Callison-Burch et al., 2007).

Translator's commentaries. As a subproduct of the human evaluation, the evaluators gave some personal impressions regarding each system performance. They concluded that, when working with the *Parliament* corpus, there was a net improvement in the overall performance when using *APE*. Changes between *RBMT* and *APE* were minor but useful. Thus, *APE* did not pose a system degradation with respect to the *RBMT*. Furthermore, a rough estimation indicated that over 10% of the sentences were perfectly translated, i.e. the translation was human-like. In addition, some frequent collocations were found to be correctly post-edited by the *APE* system, which was felt very effort saving.

With respect to the *Protocols* corpus, as expected, results were found not so satisfactory. However, human translators find themselves these documents complex.

Finally, in both cases, *APE* is able to make the translation more similar to the reference by fixing some words without altering the grammatical structure of the sentence. Finally, translators would find very useful a system that automatically decided when to automatically post-edit the *RBMT* outputs.

4 Conclusions

We have presented an automatic post-editing system that can be added at the core of the professional translation workflow. Furthermore, we have tested it with two corpora of different complexity.

For the *Parliament* corpus, we have shown that the *APE* system complements and improves the *RBMT* system in terms of suitability in a real translation scenario (average improvement 59.5%). Results for the *Protocols* corpus, although less conclusive, are promising as well (average improvement 6.5%). Moreover, 67% of *Protocols* translations, and 94% of *Parliament* translations were considered to be suitable.

Finally, a procedure for effortless human evaluation has been established. A future improvement for this would be to integrate the process in the core of the translator's workflow, so that on-the-fly evaluation can be made. In addition, several possible sources of errors have been identified which will help develop future system enhancements. For example, as stated in the translator's commentaries, the automatic selection of the most suitable translation among the systems is a desirable feature.

References

W. S. Bennett and J. Slocum. 1985. The lrc machine translation system. *Comp. Linguist.*, 11(2-3):111–121.

P. F. Brown, S. Della Pietra, V. J. Della Pietra, and R. L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Comp. Linguist.*, 19(2):263–312.

C. Callison-Burch, C. Fordyce, P. Koehn, C. Monz, and J. Schroeder. 2007. (meta-) evaluation of machine translation. In *Proc. of the 2nd Workshop on SMT*, pages 136–158, Prague, Czech Republic. ACL.

K. Chen and H. Chen. 1997. A hybrid approach to machine translation system design. In *Comp. Linguist. and Chinese Language Processing 23*, pages 241–265.

L. Dugast, J. Senellart, and P. Koehn. 2007. Statistical post-editing on SYSTRAN's rule-based translation system. In *Proc. of the 2nd Workshop on SMT*, pages 220–223, Prague, Czech Republic. ACL.

P. Isabelle, C. Goutte, and M. Simard. 2007. Domain adaptation of mt systems through automatic post-editing. In *Proc. of MTSummit XI*, pages 255–261, Copenhagen, Denmark.

P. Koehn, F. J. Och, and D. Marcu. 2003. Statistical phrase-based translation. In *Proc. of NAACL-HLT*, pages 48–54, Edmonton, Canada.

P. Koehn, 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. In *Proc. of ACL*, pages 177–180, Prague, Czech Republic.

P. Koehn. 2004. Statistical significance tests for machine translation evaluation. In *Proc. of EMNLP 2004*, Barcelona, Spain.

K. Papineni, S. Roukos, T. Ward, and W.-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proc. of ACL*, pages 311–318, Philadelphia, PA, USA.

M. Simard, C. Goutte, and P. Isabelle. 2007. Statistical phrase-based post-editing. In *Proc. of NAACL-HLT2007*, pages 508–515, Rochester, NY. ACL.

M. Snover, B. Dorr, R. Schwartz, L. Micciulla, and J. Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proc. of AMTA*, pages 223–231.

On the Importance of Pivot Language Selection for Statistical Machine Translation

Michael Paul∗†**, Hirofumi Yamamoto**†‡**, Eiichiro Sumita**† **and Satoshi Nakamura**†
† NICT, Hikaridai 2-2-2, Keihanna Science City, 619-0288 Kyoto, Japan
‡ Kinki University School of Science and Engineering, Higashi-Osaka City, 577-8502, Japan
Michael.Paul@nict.go.jp

Abstract

Recent research on multilingual statistical machine translation focuses on the usage of *pivot languages* in order to overcome resource limitations for certain language pairs. Due to the richness of available language resources, *English* is in general the pivot language of choice. In this paper, we investigate the appropriateness of languages other than English as pivot languages. Experimental results using state-of-the-art statistical machine translation techniques to translate between twelve languages revealed that the translation quality of 61 out of 110 language pairs improved when a non-English pivot language was chosen.

1 Introduction

The translation quality of state-of-the-art, phrase-based statistical machine translation (SMT) approaches heavily depends on the amount of bilingual language resources available to train the statistical models. For frequently used language pairs like *French-English* or *Chinese-English*, large-sized text data sets are readily available. There exist several data collection initiatives like the *Linguistic Data Consortium*[1], the *European Language Resource Association*[2], or the *GSK*[3], amassing and distributing large amounts of textual data. However, for less frequently used language pairs, e.g., most of the Asian languages, only a limited amount of bilingual resources are available, if at all.

In order to overcome such language resource limitations, recent research on multilingual SMT focuses on the usage of *pivot languages*. Instead of a direct translation between two languages where only a limited amount of bilingual resources is available, the *pivot translation* approach makes use of a third language that is more appropriate due to the availability of more bilingual corpora and/or its relatedness towards either the source or the target language. Several pivot translation techniques like *cascading*, *phrase-table combination*, or *pseudo corpus generation* have already been proposed (cf. Section 2).

However, for most recent research efforts, *English* is the pivot language of choice due to the richness of available language resources. For example, the Europarl corpus is exploited in (Utiyama and Isahara, 2007) for comparing pivot translation approaches between *French*, *German* and *Spanish* via *English*. Other research efforts tried to exploit the closeness between specific language pairs to generate high-quality translation hypotheses in the first step to minimize the pivot detoriation effects, e.g., for *Catalan*-to-*English* translations via *Spanish* (Gispert and Marino, 2006).

This paper investigates the appropriateness of languages other than English as pivot languages to support future research on machine translation between under-resourced language pairs. Pivot translation experiments using state-of-the-art SMT techniques are carried out to translate between twelve of the major world languages covering Indo-European as well as Asian languages and the effects of selecting a non-*English* language as the pivot language are discussed in Section 3.

2 Pivot Translation

Pivot translation is a translation from a source language (SRC) to a target language (TRG) through an intermediate *pivot* (or *bridging*) *language* (PVT). Within the SMT framework, the following coupling strategies have already been investigated:

1. *cascading of two translation systems* where the first MT engine translates the source language input into the pivot language and the second MT engine takes the obtained pivot language output as its input and translates it into the target language.
2. *pseudo corpus* approach that (i) creates a "noisy" SRC-TRG parallel corpus by translating the pivot language parts of the SRC-PVT and PVT-TRG training resources into the target language using an SMT engine trained on the PVT-TRG and PVT-SRC language resources, respectively, and (ii) directly translates the source language input into the target language using a single SMT engine that is trained on the obtained SRC-TRG language resources (Gispert and Marino, 2006).
3. *phrase-table composition* in which the translation models of the SRC-PVT and PVT-TRG translation engines are combined to a new SRC-TRG phrase-table by merging SRC-PVT and PVT-TRG phrase-table entries with identical pivot language phrases and mul-

[1]LDC: http://www.ldc.upenn.edu
[2]ELRA: http://www.elra.info
[3]GSK: http://www.gsk.or.jp/catalog.html

Proceedings of NAACL HLT 2009: Short Papers, pages 221–224,
Boulder, Colorado, June 2009.

tiplying posterior probabilities (Utiyama and Isahara, 2007; Wu and Wang, 2007).

4. *bridging at translation time* where the coupling is integrated into the SMT decoding process by modeling the pivot text as a hidden variable and assuming independence between source and target sentences (Bertoldi et al., 2008).

3 Pivot Language Selection

The effects of using different pivot languages are investigated using the multilingual *Basic Travel Expressions Corpus* (BTEC), which is a collection of sentences that bilingual travel experts consider useful for people going to or coming from another country. For the pivot translation experiments, we selected twelve of the major world languages covered by BTEC, favoring languages that are actively being researched on, i.e.,*Chinese* (zh), *English* (en), *French* (fr), *German* (de), *Hindi* (hi), *Indonesian* (id), *Japanese* (ja), *Korean* (ko), *Malay* (ms), *Spanish* (es), *Thai* (th), and *Vietnamese* (vi). These languages differ largely in *word order* (SVO, SOV), *segmentation unit* (phrase, word, none), and *degree of inflection* (high, moderate, light). All data sets were case-sensitive with punctuation marks preserved.

However, in a real-world application, identical language resources covering three or more languages are not necessarily to be expected. In order to avoid a trilingual scenario for the pivot translation experiments described in this paper, the 160k sentence-aligned BTEC corpus was randomly split into two subsets of 80k sentences each, whereby the first set of sentence pairs was used to train the source-to-pivot translation models ($80k^{sp}$) and the second subset of sentence pairs was used to train the pivot-to-target translation models ($80k^{pt}$). Table 1 summarizes the characteristics of the BTEC corpus data sets used for the training (*train*) of the SMT models, the tuning of model weights (*dev*), and the evaluation of translation quality (*eval*). Besides the number of sentences (*sen*) and the vocabulary (*voc*), the sentence length (*len*) is also given, as the average number of words per sentence.

For the training of the SMT models, standard word alignment (Och and Ney, 2003) and language modeling (Stolcke, 2002) tools were used. Minimum error rate training (MERT) was used to tune the decoder's parameters, and performed on the *dev* set using the technique proposed in (Och and Ney, 2003). For the translation, an in-house multi-stack phrase-based decoder comparable to MOSES was used. For the evaluation of translation quality, we applied standard automatic evaluation metrics, i.e., BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005). For the experimental results in this paper, the given scores are calculated as the average of the respective BLEU and METEOR scores obtained for each system output and are listed as percent figures.

Table 1: Language Resources

BTEC Corpus		train 80k^{sp}	train 80k^{pt}	dev set	eval set
# of sen		80,000	80,000	1,000	1,000
en	voc	12,264	11,047	1,262	1,292
	len	7.8	7.2	7.1	7.2
de	voc	19,593	17,324	1,486	1,491
	len	7.4	6.8	6.7	6.8
es	voc	16,317	14,807	1,486	1,511
	len	7.6	7.1	7.0	7.2
fr	voc	15,319	13,663	1,455	1,466
	len	7.8	7.3	7.1	7.3
hi	voc	26,096	19,906	1,558	1,588
	len	8.1	7.6	7.4	7.5
id	voc	14,585	13,224	1,433	1,394
	len	7.0	6.5	6.3	6.4
ja	voc	13,868	12,517	1,407	1,408
	len	8.8	8.2	8.1	8.2
ko	voc	13,546	12,281	1,366	1,365
	len	8.3	7.8	7.7	7.8
ms	voc	15,113	13,616	1,459	1,438
	len	7.1	6.6	6.4	6.5
th	voc	6,103	5,603	1,081	1,053
	len	8.1	7.4	7.3	7.4
vi	voc	7,980	7,335	1,245	1,267
	len	9.4	8.7	8.5	8.6
zh	voc	11,084	10,159	1,312	1,301
	len	7.1	6.6	6.4	6.5

In order to get an idea of how difficult the translation task for the different languages is supposed to be, the automatic evaluation scores for the direct translation approach using the $80k^{sp}$ language resources are summarized in Section 3.1. The effects of the pivot language selection are discussed in Section 3.2 using the pivot translation method of *cascading two SMT systems*. In addition, the dependency between selecting the optimal pivot language for a given language pair and the amount of available training resources are described in Section 3.3.

3.1 Direct Translation Results

The automatic evaluation scores for all source and target language pair combinations of the direct translation approach are given in Table 2. For each target language, the highest evaluation scores are marked in boldface and the lowest scores are marked in typewriter mode.

The highest translation quality was achieved for the *Japanese* $\Leftrightarrow$ *Korean*, *Indonesian* $\Leftrightarrow$ *Malay*, and *Spanish* $\Leftrightarrow$ *English* translation tasks. In addition, relatively high evaluation scores were achieved for *Japanese* $\Leftrightarrow$ *Chinese* and for translations from *English* into *German*, *French*, *Hindi*, *Thai*, and *Vietnamese*. On the other hand, the most difficult translation tasks were those having *Korean* or *Chinese* as the source language.

3.2 Pivot Translation Results

The automatic evaluation scores for all pivot translation combinations are summarized in Table 3 whereby for each source-target language pair, the results of the experiments using (i) *English* (*en*) and (ii) the best performing language (*best*) as the pivot language are listed.

Comparing the results of the pivot translation experiments towards the direct translation results, we can see

Table 2: Translation Quality of Direct Translation Approach

SRC	de	en	es	fr	hi	id	ja	ko	ms	th	vi	zh
de	–	74.24	56.22	49.78	63.25	69.31	54.09	50.88	69.33	66.83	67.17	51.59
en	**63.31**	–	**64.30**	**56.10**	**66.43**	73.46	55.64	54.15	73.66	**70.57**	**72.64**	53.18
es	58.98	**76.43**	–	53.53	63.60	70.46	55.37	51.41	70.46	67.69	69.15	52.03
fr	55.45	72.24	57.25	–	61.70	68.58	55.17	52.15	68.72	65.03	65.97	52.83
hi	52.89	67.82	50.69	45.53	–	68.65	52.94	50.93	68.14	66.44	66.88	51.31
id	52.75	67.58	52.06	46.00	62.43	–	55.52	52.90	**88.69**	67.20	68.01	52.77
ja	35.43	51.65	37.82	32.70	46.94	52.90	–	**78.73**	53.26	54.14	51.45	**67.83**
ko	32.65	50.12	36.97	31.62	44.67	53.51	**78.88**	–	51.75	52.35	51.34	63.19
ms	53.16	68.17	53.06	45.30	63.36	**91.12**	54.88	52.18	–	67.79	67.93	53.23
th	49.66	64.53	50.16	42.70	59.40	66.58	53.82	50.81	65.76	–	65.90	52.22
vi	52.59	69.16	53.17	45.60	61.19	68.39	52.95	50.68	69.44	67.64	–	51.29
zh	34.18	49.79	37.13	31.16	44.33	52.72	65.64	62.23	52.46	51.88	51.09	–

Table 4: Pivot Language Selection

PVT	usage (%)	
en	49	(44.5)
ms	16	(14.5)
id	16	(14.5)
ja	14	(12.7)

PVT	usage (%)	
ko	12	(10.9)
zh	2	(1.8)
es	1	(0.9)

that in general the pivot translation approach performs worse than the direct translation approach due to the effect of error chaining, i.e., translation errors of the SRC-PVT engine cause a degradation in translation quality of the PVT-TRG system output. However, for language pairs like *Korean* ⇔ *German*, *Japanese* ⇔ *Indonesian* and *German/Spanish* ⇔ *Korean*, the best pivot translation system outperforms the direct translation approach slightly. This phenomenon is caused mainly by the high SRC-PVT (*PVT-TRG*) translation quality in combination with a better *PVT-TRG* (SRC-PVT) performance compared to the direct *SRC-TRG* system output results.

Besides the automatic evaluation scores, Table 3 lists also the optimal pivot language for each source-target language pair in boldface. The experimental results show that *English* is indeed the best pivot language when translating between languages, like *German*, *Spanish*, *French*, *Hindi*, *Thai*, and *Vietnamese*, whose direct translation performance from/into *English* is high. For these six languages, all language pair combinations achieved the highest scores using the *English* pivot translation approach. In contrast, *English* is the pivot language of choice for only 16.2% (11 out of 68) of the language pairs when translating from/into *Japanese*, *Korean*, *Indonesian*, or *Malay*. In the remaining cases, the language with the highest direct translation scores is in general selected as the optimal pivot language, i.e., *Japanese* for *Korean*, *Malay* for *Indonesian* and vice versa. For *Chinese*, the choice of the optimal pivot language varies largely depending on the language direction. However, the selection of the optimal pivot language is not symmetric for 34.5% of the language pairs, i.e., a different optimal pivot language was obtained for the SRC-TRG compared to the TRG-SRC translation task. This indicates that the choice of the optimal pivot language depends on the relatedness of the SRC and PVT languages as well as the relatedness of the PVT and TRG languages.

The distribution of the optimal pivot language selection for all language pairs is given in Table 4. The figures show that the *English* pivot approach still achieves the highest scores for the majority of the examined language pairs. However, in 55.5% (61 out of 110) of the cases, a non-English pivot language, mainly *Malay*, *Indonesian*, *Japanese*, or *Korean*, is to be preferred.

Table 5: Pivot Selection Shifts for 10k vs. 80k Training Data

10k PVT	80k PVT	pivot translation language pair
ko	**en**	ja-**fr**, ja-**de**, ja-**vi**
ko		zh-**fr**, zh-**es**, zh-**hi**
ja		ko-**vi**, zh-**vi**, zh-**th**
ms		id-**fr**
ja	**es**	ko-**hi**
ja	**id**	ko-**ms**,th-**zh**
en		es-**ms**,hi-**zh**,hi-**ja**

10k PVT	80k PVT	pivot translation language pair
ko	**ms**	ja-**id**
en		vi-**zh**
es		fr-**zh**
en	**ja**	fr-**ko**, hi-**ko**, vi-**ko**
id		zh-**ms**
ms	**ko**	id-**ja**
es		fr-**ja**
en		de-**ja**,es-**ja**

3.3 Training Data Size Dependency

In order to investigate the dependency between selecting the optimal pivot language for a given language pair and the amount of available training resources, we repeated the pivot translation experiments described in Section 3.2 for statistical models trained on subsets of 10k sentences randomly extracted from the $80k^{sp}$ and the $80k^{pt}$ corpora, respectively.

The results showed that 75.5% of the pivot language selections are identical for small (10k) and large (80k) training data sets. For the remaining 27 out of 110 translation tasks, Table 5 lists how the optimal pivot language selection changed. In the case of small training data sets, the pivot language is closely related (in terms of high direct translation quality) to the source language. However, for larger training data sets, the focus shifts towards closely related target languages. Therefore, the higher the translation quality of the pivot translation task is, the more dependend the selection of the optimal pivot language is on the system performance of the PVT-TRG task.

4 Conclusion

In this paper, the effects of using non-English pivot languages for translations between twelve major world languages were compared to the standard English pivot translation approach. The experimental results revealed that *English* was indeed more frequently (45.5% out of

Table 3: Translation Quality of Pivot Translation Approach

SRC	PVT	de	es	fr	hi	id	ja	ko	ms	th	vi	zh
de	en	–	54.69	47.01	60.48	66.42	52.53	51.10	66.47	65.06	66.08	50.46
	best	–	(en) 54.69	(en) 47.01	(en) 60.48	(**ms**) **66.92**	(**ko**) **52.67**	(en) 51.10	(en) 66.47	(en) 65.06	(en) 66.08	(en) 50.46
es	en	55.37	–	48.75	60.24	68.10	52.68	51.80	67.54	65.59	66.99	51.08
	best	(en) 55.37	–	(en) 48.75	(en) 60.24	(**ms**) **69.29**	(**ko**) **53.10**	(en) 51.80	(**id**) **68.37**	(en) 65.59	(en) 66.99	(en) 51.08
fr	en	52.03	53.88	–	58.27	65.59	52.51	51.19	65.43	62.47	64.34	50.12
	best	(en) 52.03	(en) 53.88	–	(en) 58.27	(**ms**) **67.25**	(**ko**) **53.06**	(**ja**) **51.81**	(en) 65.43	(en) 62.47	(en) 64.34	(**ms**) **50.35**
hi	en	48.56	48.69	41.71	–	63.01	50.21	48.96	63.13	62.08	62.48	48.12
	best	(en) 48.56	(en) 48.69	(en) 41.71	–	(**ms**) **65.43**	(**id**) **51.09**	(**ja**) **49.06**	(**id**) **65.54**	(en) 62.08	(en) 62.48	(**id**) **48.71**
id	en	48.97	49.48	42.56	57.41	–	51.30	50.19	72.94	62.40	64.60	49.45
	best	(**ms**) **49.19**	(**ms**) **50.16**	(en) 42.56	(**ms**) **60.30**	–	(**ko**) **54.12**	(**ja**) **51.54**	(en) 72.94	(**ms**) **64.51**	(**ms**) **65.51**	(**ms**) **51.82**
ja	en	33.43	36.61	31.20	44.27	52.31	–	56.34	51.34	52.57	50.97	52.85
	best	(en) 33.43	(**ko**) **36.88**	(en) 31.20	(**ko**) **44.96**	(**ms**) **53.13**	–	(**zh**) **60.99**	(**ko**) **51.37**	(**ko**) **52.65**	(en) 50.97	(**ko**) **62.65**
ko	en	31.52	34.50	29.01	43.23	50.70	54.43	–	49.83	50.74	49.97	51.66
	best	(**ja**) **33.23**	(**ja**) **36.18**	(**ja**) **31.20**	(**es**) **44.24**	(**ja**) **52.21**	(**zh**) **60.10**	–	(**id**) **51.79**	(**ja**) **51.98**	(en) 49.97	(**ja**) **62.74**
ms	en	49.64	49.71	42.39	57.85	73.25	51.01	49.52	–	62.64	64.09	49.22
	best	(**id**) **51.14**	(**id**) **50.95**	(**id**) **43.87**	(**id**) **60.76**	(en) 73.25	(**id**) **54.56**	(**ja**) **50.94**	–	(**id**) **65.99**	(**id**) **66.97**	(**id**) **52.46**
th	en	46.57	46.61	39.83	55.41	61.88	50.54	48.75	61.09	–	61.50	47.75
	best	(en) 46.57	(en) 46.61	(en) 39.83	(en) 55.41	(**ms**) **63.22**	(**ko**) **51.37**	(**ja**) **50.39**	(**id**) **62.36**	–	(en) 61.50	(**id**) **48.72**
vi	en	49.87	50.17	43.04	57.42	64.94	50.68	49.45	64.60	62.50	–	48.12
	best	(en) 49.87	(en) 50.17	(en) 43.04	(en) 57.42	(**ms**) **67.14**	(**ko**) **51.86**	(**ja**) **49.48**	(**id**) **66.57**	(en) 62.50	–	(**ms**) **48.86**
zh	en	32.26	35.29	28.35	43.20	50.11	53.27	52.53	49.20	51.54	49.92	–
	best	(en) 32.26	(en) 35.29	(en) 28.35	(en) 43.20	(**ms**) **52.18**	(**ko**) **61.96**	(**ja**) **60.64**	(**ja**) **49.71**	(en) 51.54	(en) 49.92	–

110 language pairs) selected as the best pivot language over any other examined language. However, its usage is limited to translations between Indo-European languages and some Asian languages like *Thai* or *Vietnamese*. Otherwise, the *English* pivot approach is largely outperformed by using Asian languages as the pivot languages, especially *Japanese*, *Malay*, *Indonesian*, or *Korean*.

The analysis of the results revealed that the selection of the optimal pivot language largely depends on the SRC-PVT and PVT-TRG translation performance, i.e., for small training corpora, the relationship between source/pivot languages seems to be more important, whereas the selection criteria moves towards the relationship between pivot/target languages for larger amounts of training data and thus for MT engines of higher translation quality.

In order to explore the question of pivot selection further and arrive at firmer conclusions, future work will have to investigate in detail what kind of features are important in selecting a pivot language for a given language pair. Besides the translation quality of SMT engines, automatic metrics to measure the relatedness of a language pair should also be taken into account to find optimal pivot languages. For example, (Birch et al., 2008) proposes features like *amount of reordering*, *the morphological complexity of the target language*, and *historical relatedness of the two languages* as strong predictors for the variability of SMT system performance.

In addition, concerning the question of how the pivot language selection criteria depends on the choice of the pivot translation method, future work will also have to investigate the effects of pivot language selection for the other pivot translation approaches described in Section 2.

Based on these findings, we plan to determine the contribution of different language characteristics on the system performance automatically to obtain useful indicators that could be used to train statistical classification models to predict the best pivot language for a new language pair and improve the usability of machine translation between under-resourced languages further.

Acknowledgment

This work is partly supported by the Grant-in-Aid for Scientific Research (C) Number 19500137 and "Construction of speech translation foundation aiming to overcome the barrier between Asian languages", the Special Coordination Funds for Promoting Science and Technology of the Ministry of Education, Culture, Sports, Science and Technology, Japan.

References

S. Banerjee and A. Lavie. 2005. METEOR: An Automatic Metric for MT Evaluation. In *Proc. of the ACL*, pages 65–72, Ann Arbor, US.

N. Bertoldi, M. Barbaiani, M. Federico, and R. Cattoni. 2008. Phrase-Based SMT with Pivot Languages. In *Proc. of the IWSLT*, pages 143–149, Hawaii, US.

A. Birch, M. Osborne, and P. Koehn. 2008. Predicting Success in MT. In *Proc. of the EMNLP*, pages 744–753, Hawaii, US.

A. Gispert and J. Marino. 2006. Catalan-English SMT without Parallel Corpus: Bridging through Spanish. In *Proc. of 5th LREC*, pages 65–68, Genoa, Italy.

F. Och and H. Ney. 2003. A Systematic Comparison of Statistical Alignment Models. *Computational Linguistics*, 29(1):19–51.

K. Papineni, S. Roukos, T. Ward, and W. Zhu. 2002. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proc. of the 40th ACL*, pages 311–318, Philadelphia, US.

A. Stolcke. 2002. SRILM an extensible language modeling toolkit. In *Proc. of ICSLP*, pages 901–904, Denver.

M. Utiyama and H. Isahara. 2007. A comparison of pivot methods for phrase-based SMT. In *Proc. of HLT*, pages 484–491, New York, US.

H. Wu and H. Wang. 2007. Pivot Language Approach for Phrase-Based SMT. In *Proc. of ACL*, pages 856–863, Prague, Czech Republic.

Tree Linearization in English: Improving Language Model Based Approaches

Katja Filippova and **Michael Strube**
EML Research gGmbH
Schloss-Wolfsbrunnenweg 33
69118 Heidelberg, Germany
http://www.eml-research.de/nlp

Abstract

We compare two approaches to dependency tree linearization, a task which arises in many NLP applications. The first one is the widely used 'overgenerate and rank' approach which relies exclusively on a trigram language model (LM); the second one combines language modeling with a maximum entropy classifier trained on a range of linguistic features. The results provide strong support for the combined method and show that trigram LMs are appropriate for phrase linearization while on the clause level a richer representation is necessary to achieve comparable performance.

1 Introduction

To date, many natural language processing applications rely on syntactic representations and also modify them by compressing, fusing, or translating into a different language. A syntactic tree emerging as a result of such operations has to be linearized to a string of words before it can be output to the end-user. The simple and most widely used trigram LM has become a standard tool for tree linearization in English (Langkilde & Knight, 1998). For languages with less rigid word order, LM-based approaches have been shown to perform poorly (e.g., Marsi & Krahmer (2005) for Dutch), and methods relying on a range of linguistic features have been successfully applied instead (see Uchimoto et al. (2000) and Ringger et al. (2004), Filippova & Strube (2007) for Japanese and German resp.). To our knowledge, none of the linearization studies have compared a LM-based method with an alternative. Thus, it would be of interest to draw such a comparison, especially on English data, where LMs are usually expected to work well.

As an improvement to the LM-based approach, we propose a combined method which distinguishes between the phrase and the clause levels:

- it relies on a trigram LM to order words within phrases;
- it finds the order of clause constituents (i.e., constituents dependent on a finite verb) with a maximum entropy classifier trained on a range of linguistic features.

We show that such a differentiated approach is beneficial and that the proposed combination outperforms the method which relies solely on a LM. Hence, our results challenge the widespread attitude that trigram LMs provide an appropriate way to linearize syntactic trees in English but also indicate that they perform well in linearizing subtrees corresponding to phrases.

2 LM-based Approach

Trigram models are easy to build and use, and it has been shown that more sophisticated n-gram models (e.g., with higher n, complex smoothing techniques, skipping, clustering or caching) are often not worth the effort of implementing them due to data sparseness and other issues (Goodman, 2001). This explains the popularity of trigram LMs in a variety of NLP tasks (Jurafsky & Martin, 2008), in particular, in tree linearization where they have become

Proceedings of NAACL HLT 2009: Short Papers, pages 225–228,
Boulder, Colorado, June 2009.

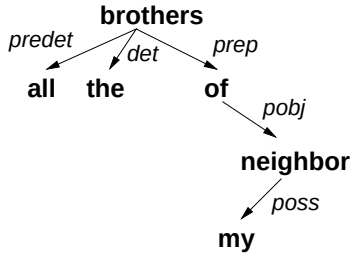


Figure 1: A tree of the noun phrase *all the brothers of my neighbor*

de facto the standard tree linearization tool in accordance with the 'overgenerate and rank' principle: given a syntactic tree, one needs to consider all possible linearizations and then choose the one with the lowest entropy. Given a projective dependency tree[1], all linearizations can be found recursively by generating permutations of a node and its children. Unfortunately, the number of possible permutations grows factorially with the branching factor. Hence it is highly desirable to prohibit generation of clearly unacceptable permutations by putting hard constraints encoded in the English grammar. The constraints which we implement in our study are the following: determiners, possessives, quantifiers and noun or adjective modifiers always **precede** their heads. Conjunctions, coordinated elements, prepositional objects always **follow** their heads. These constraints allow us to limit, e.g., the total of 96 ($2 \times 2 \times 4!$) possibilities for the tree corresponding to the phrase *all the brothers of my neighbor* (see Figure 1) to only two (*all the brothers of my neighbor, the all brothers of my neighbor*).

Still, even with such constraints, in some cases the list of possible linearizations is too long and has to be reduced to the first N, where N is supposed to be sufficiently large. In our experiments we break the permutation generation process if the limit of 20,000 variants is reached.

3 Combined Approach

The LM approach described above has at least two disadvantages: (1) long distance dependencies are not captured, and (2) the list of all possible linearizations can be huge which makes the search for the best string unfeasible. However, our combined approach is based on the premise that trigram LMs are well-suited for finding the order within NPs, PPs and other phrases where the head is not a finite verb. E.g., given a noun modified by the words *big*, *red* and *the*, a LM can reliably rank the correct order higher than incorrect ones (*the big red N* vs. *the red big N*, etc.).

Next, on the clause level, for every finite verb in the tree we find the order of its dependents using the method which we originally developed for German (Filippova & Strube, 2007), which utilizes a range of such linguistic features as PoS tag, syntactic role, length in words, pronominalization, semantic class, etc.[2] For the experiments presented in this paper, we train two maximum entropy classifiers on all but the semantic features:

1. The first classifier determines the best starting point for a sentence: for each constituent dependent on the verb it returns the probability of this constituent being the first one in a sentence. The subject and also adjuncts (e.g. temporal adjuncts like *yesterday*) are usually found in the beginning of the sentence.

2. The second classifier is trained to determine whether the precedence relation holds between two adjacent constituents and is applied to all constituents but the one selected by the first classifier. The precedence relation defined by this classifier has been shown to be transitive and thus can be used to sort randomly ordered constituents. Note that we do not need to consider all possible orders to find the best one.

Once the order within clause constituents as well as the order among them is found, the verb is placed right after the subject. The verb placing step completes the linearization process.

The need for two distinct classifiers can be illustrated with the following example:

(1) a [*Earlier today*] [she] sent [him] [an email].

b [She] sent [him] [an email] [*earlier today*].

c *[She] sent [*earlier today*] [him] [an email].

[1]Note that a phrase structure tree can be converted into a dependency tree, and some PCFG parsers provide this option.

[2]See the cited paper for the full list of features and implementation details.

(1a,b) are grammatical while (1c) is hardly acceptable, and no simple precedence rule can be learned from pairs of constituents in (1a) and (1b): the temporal adjunct *earlier today* can precede or follow each of the other constituents dependent on the verb (*she, him, an email*). Thus, the classifier which determines the precedence relation is not enough. However, an adequate rule can be inferred with an additional classifier trained to find good starting points: a temporal adjunct may appear as the first constituent in a sentence; if it is not chosen for this position, it should be preceded by the pronominalized subject (*she*), the indirect object (*him*) and the short non-pronominalized object (*an email*).

4 Experiments

The goal of our experiments is to check the following hypotheses:

1. That trigram LMs are well-suited for phrase linearization.
2. That there is a considerable drop in performance when one uses them for linearization on the clause level.
3. That an approach which uses a richer representation on the clause level is more appropriate.

4.1 Data

We take a subset of the TIPSTER[3] corpus – all Wall Street Journal articles from the period of 1987-92 (approx. 72 mill. words) – and automatically annotate them with sentence boundaries, part of speech tags and dependency relations using the Stanford parser (Klein & Manning, 2003). We reserve a small subset of about 600 articles (340,000 words) for testing and use the rest to build a trigram LM with the CMU toolkit (Clarkson & Rosenfeld, 1997, with Good-Turing smoothing and vocabulary size of 30,000). To train the maximum entropy classifiers we use about 41,000 sentences.

4.2 Evaluation

To test the trigram-based approach, we generate all possible permutations of clause constituents, place the verb right after the subject and then rank the resulting strings with the LM taking the information on sentence boundaries into account. To test the combined approach, we find the best candidate for the first position in the clause, then put the remaining constituents in a random order, and finally sort them by consulting the second classifier.

The purpose of the evaluation is to assess how good a method is at reproducing the input from its dependency tree. We separately evaluate the performance on the phrase and the clause levels. When comparing the two methods on the clause level, we take the clause constituents as they are presented in the input sentence. Although English allows for some minor variation in word order and it might happen that the generated order is not necessarily wrong if different from the original one, we do not expect this to happen often and evaluate the performance rigorously: only the original order counts as the correct one. The default evaluation metric is per-phrase/per-clause accuracy:

$$acc = \frac{|correct|}{|total|}$$

Other metrics we use to measure how different a generated order of N elements is from the correct one are:

1. Kendall's τ, $\tau = 1 - 4\frac{t}{N(N-1)}$ where t is the minimum number of interchanges of consecutive elements to achieve the right order (Kendall, 1938; Lapata, 2006).
2. Edit distance related di, $di = 1 - \frac{m}{N}$ where m is the minimum number of deletions combined with insertions to get to the right order (Ringger et al., 2004).

E.g., on the phrase level, the incorrectly generated phrase *the all brothers of my neighbor* ('1-0-2-3-4-5') gets $\tau = 0.87$, $di = 0.83$. Likewise, given the input sentence from (1a), the incorrectly generated order of the four clause constituents in (1c) – '1-0-2-3' – gets τ of 0.67 and di of 0.75.

4.3 Results

The results of the experiments on the phrase and the clause levels are presented in Tables 1 and 2 respectively. From the total of 5,000 phrases, 55 (about

[3]Description at `http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC93T3A`.

1%) were discarded because the number of admissible linearizations exceeded the limit of 20,000. In the first row of Table 1 we give the results for cases where, with all constraints applied, there were still several possible linearizations (*non-triv*; 1,797); the second row is for all phrases which were longer than one word (> 1; 2,791); the bottom row presents the results for the total of 4,945 phrases (*all*).

	acc	τ	*di*
non-triv	76%	0.85	0.94
> 1	85%	0.90	0.96
all	91%	0.94	0.98

Table 1: Results of the trigram method on the phrase level

Table 2 presents the results of the trigram-based (TRIGRAM) and combined (COMBINED) methods on the clause level. Here, we filtered out trivial cases and considered only clauses which had at least two constituents dependent on the verb (approx. 5,000 clauses in total).

	acc	τ	*di*
TRIGRAM	49%	0.49	0.81
COMBINED	67%	0.71	0.88

Table 2: Results of the two methods on the clause level

4.4 Discussion

The difference in accuracy between the performance of the trigram model on the phrase and the clause level is considerable – 76% vs. 49%. The accuracy of 76% is remarkable given that the average length of phrases which counted as *non-triv* is 6.2 words, whereas the average clause length in constituents is 3.3. This statistically significant difference in performance supports our hypothesis that the 'overgenerate and rank' approach advocated in earlier studies is more adequate for finding the optimal order within phrases. The τ value of 0.85 also indicates that many of the wrong phrase linearizations were near misses. On the clause level, where long distance dependencies are frequent, an approach which takes a range of grammatical features into account is more appropriate – this is confirmed by the significantly better results of the combined method (67%).

5 Conclusions

We investigated two tree linearization methods in English: the mainstream trigram-based approach and the one which combines a trigram LM on the phrase level with two classifiers trained on a range of linguistic features on the clause level. The results demonstrate (1) that the combined approach reproduces the word order more accurately, and (2) that the performance of the trigram LM-based method on phrases is significantly better than on clauses.

Acknowledgments: This work has been funded by the Klaus Tschira Foundation, Heidelberg, Germany. The first author has been supported by a KTF grant (09.009.2004). We would like to thank the anonymous reviewers for their feedback.

References

Clarkson, P. & R. Rosenfeld (1997). Statistical language modeling using the CMU-Cambridge toolkit. In *Proc. of EUROSPEECH-97*, pp. 2707–2710.

Filippova, K. & M. Strube (2007). Generating constituent order in German clauses. In *Proc. of ACL-07*, pp. 320–327.

Goodman, J. T. (2001). A bit of progress in language modeling. *Computer Speech and Language*, pp. 403–434.

Jurafsky, D. & J. H. Martin (2008). *Speech and Language Processing.* Upper Saddle River, N.J.: Prentice Hall.

Kendall, M. G. (1938). A new measure of rank correlation. *Biometrika*, 30:81–93.

Klein, D. & C. D. Manning (2003). Accurate unlexicalized parsing. In *Proc. of ACL-03*, pp. 423–430.

Langkilde, I. & K. Knight (1998). Generation that exploits corpus-based statistical knowledge. In *Proc. of COLING-ACL-98*, pp. 704–710.

Lapata, M. (2006). Automatic evaluation of information ordering: Kendall's tau. *Computational Linguistics*, 32(4):471–484.

Marsi, E. & E. Krahmer (2005). Explorations in sentence fusion. In *Proc. of ENLG-05*, pp. 109–117.

Ringger, E., M. Gamon, R. C. Moore, D. Rojas, M. Smets & S. Corston-Oliver (2004). Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proc. of COLING-04*, pp. 673–679.

Uchimoto, K., M. Murata, Q. Ma, S. Sekine & H. Isahara (2000). Word order acquisition from corpora. In *Proc. of COLING-00*, pp. 871–877.

Determining the Position of Adverbial Phrases in English

Huayan Zhong and **Amanda Stent**
Computer Science Department
Stony Brook University
Stony Brook, NY 11794, USA
zhong@cs.sunysb.edu, amanda.stent@stonybrook.edu

Abstract

In this paper we compare three approaches to adverbial positioning using lexical, syntactic, semantic and sentence-level features. We find that: (a), one- and two-stage classification-based approaches can achieve almost 86% accuracy in determining the absolute position of adverbials; (b) a classifier trained with only syntactic features gives performance close to that of a classifier trained with all features; and (c) a surface realizer incorporating a two-stage classifier for adverbial positioning as the second stage gives improvements of at least 10% in simple string accuracy over a baseline realizer for sentences containing adverbials.

1 Introduction

The job of a *surface realizer* is to transform an input semantic/syntactic form into a sequence of words. This task includes word choice, and word and constituent ordering. In English, the positions of required elements of a sentence, verb phrase or noun phrase are relatively fixed. However, many sentences also include *adverbials* whose position is not fixed (Figure 1). There may be several appropriate positions for an adverbial in a particular context, but other positions give output that is non-idiomatic or disfluent, ambiguous, or incoherent.

Some computational research has included models for adjunct ordering (e.g. (Ringger et al., 2004; Marciniak and Strube, 2004; Elhadad et al., 2001)). However, this is the first computational study to look specifically at adverbials. Adverbial positioning has long been studied in linguistics (e.g. (Keyser, 1968; Allen and Cruttenden, 1974; Ernst, 1984; Haider, 2000)). Most linguistic research focuses on whether adverbial placement is functional or semantic in nature. However, Costa (2004) takes a more flexible feature-based approach that uses: lexical features (e.g. phonological shape, ambiguity of meaning, categorical status); syntactic features (e.g. possible adjunction sites, directionality of adjunction, domain of modification); and information structure features (e.g. focus, contrast). We decided to evaluate Costa's approach computationally, using features automatically extracted from an annotated corpus.

In this paper, we compare three approaches to adverbial positioning: a simple baseline approach using lexical and syntactic features, and one- and two-stage classification-based approaches using lexical, syntactic, semantic and sentence-level features. We apply these approaches in a hybrid surface realizer that uses a probabilistic grammar to produce realization alternatives, and a second-stage classifier to select among alternatives. We find that: (a) One- and two-stage classification-based approaches can achieve almost 86% accuracy in determining the absolute position of adverbials; (b) A classifier trained with only syntactic features gives performance close to that of a classifier trained with all features; and (c) A surface realizer using a two-stage classifier for adverbial positioning can get improvements of at least 10% in simple string accuracy over a baseline realizer for sentences containing adverbials.

As well as being useful for surface realization, a model of adverbial ordering can be used in machine translation (e.g. (Ogura et al., 1997)), language learning software (e.g. (Leacock, 2007; Burstein et al., 2004)), and automatic summarization (e.g. (Elhadad et al., 2001; Clarke and Lapata, 2007; Madnani et al., 2007)).

Proceedings of NAACL HLT 2009: Short Papers, pages 229–232,
Boulder, Colorado, June 2009.

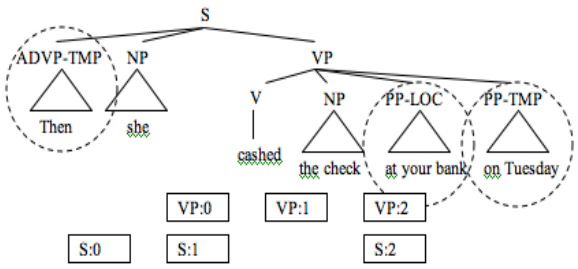


Figure 1: Example syntax tree for *Then she cashed the check at your bank on Tuesday* with adverbials circled and possible VP and S adverbial positions in squares.

2 Data and Features

From the sentences in the Wall Street Journal (WSJ) and Switchboard (SWBD) sections of the Penn Treebank III (Marcus et al., 1999), we extracted all NP, PP and ADVP phrases labeled with the adverbial tags -BNF, -DIR, -EXT, -LOC, -MNR, -PRP, -TMP or -ADV. These phrases mostly modify S constituents (including RRC, S, SBAR, SBARQ, SINV, SQ), VP constituents, or NP constituents (including NP and WHNP), but also modify other adjuncts (PP, ADJP or ADVP) and other phrase types (FRAG, INTJ, LST, NAC, PRT, QP, TOP, UCP, X).

Corpus	Number of adverbials of type:		
	PP-ADVP	NP-ADVP	ADVP
WSJ	36128	10587	13700
SWBD	12231	5405	17193

Table 1: Adverbials in the Penn Treebank III

For each adverbial, we automatically extracted lexical, syntactic, semantic and discourse features. We included features similar to those in (Costa, 2004) and from our own previous research on prepositional phrase ordering (Zhong and Stent, 2008). Due to the size of our data set, we could only use features that can be extracted automatically, so some features were approximated. We dropped adverbials for which we could not get features, such as empty adverbials. Tables 1 and 2 summarize the resulting data. A list of the features we used in our classification experiment appears in Table 3. We withheld 10% of this data for our realization experiment.

3 Classification Experiment

Our goal is to determine the position of an adverbial with respect to its siblings in the phrase of which it is a part. An adverbial may have non-adverbial siblings, whose position is typically fixed. It may also have other adverbial siblings. In the sentence in Figure 1, *at your bank* has one adverbial and two non-adverbial siblings. If this adverbial were placed at positions VP:0 or VP:1 the resulting sentence would be disfluent but meaningful; placed at position VP:2 the resulting sentence is fluent, meaningful and idiomatic. (In this sentence, both orderings of the two adverbials at position VP:2 are valid.)

Adverbial Type	Data Set	
	WSJ	SWBD
S	8196	5144
VP	29734	22845
NP	12985	2071
PP/ADJP/ADVP	1739	987
Other	297	686

Table 2: Adverbials in the Penn Treebank III

3.1 Approaches

We experimented with three approaches to adverbial positioning.

Baseline Our baseline approach has two stages. In the first stage the position of each adverbial with respect to its non-adverbial siblings is determined: each adverbial is assigned the most likely position given its lexical head and category (PP, NN, ADVP). In the second stage, the relative ordering of adjacent adverbials is determined in a pairwise fashion (cf. (Marciniak and Strube, 2004)): the ordering of a pair of adverbials is assigned to be the most frequent in the training data, given the lexical head, adverbial phrase type, and category of each adverbial.

One-stage For our one-stage classification-based approach, we determine the position of all adverbials in a phrase at one step. There is one feature vector for each phrase containing at least one adverbial. It contains features for all non-adverbial siblings in realization order, and then for each adverbial sibling in alphabetical order by lexical head. The label is the order of the siblings. For example, for the S-modifying adverbial in Figure 1, the label would be 2_0_1, where 0 = "she", 1 = "cashed" and 2 = "Then". If there are n siblings, then there are $n!$ possible labels for each feature vector, so the performance of this classifier by chance would be .167 if each adverbial has on average three siblings.

Type	Features
lexical	preposition in this adverbial and in adverbial siblings 0-4; stems of lexical heads of this adverbial, its parent, non-adverbial siblings 0-4, and adverbial siblings 0-4; number of phonemes in lexical head of this adverbial and in lexical heads of adverbial siblings 0-4; number of words in this adverbial and in adverbial siblings 0-4
syntactic	syntactic categories of this adverbial, its parent, non-adverbial siblings 0-4, and adverbial siblings 0-4; adverbial type of this adverbial and of adverbial siblings 0-4 (one of DIR, EXT, LOC, MNR, PRP, TMP, ADV); numbers of siblings, non-adverbial siblings, and adverbial siblings
semantic	hypernyms of heads of this adverbial, its parent, non-adverbial siblings 0-4, and adverbial siblings 0-4; number of meanings for heads of this adverbial and adverbial siblings 0-4 (using WordNet)
sentence	sequence of children of S node (e.g. NP VP, VP); form of sentence (declarative, imperative, interrogative, clause-other); presence of the following in the sentence: coordinating conjunction(s), subordinating conjunction(s), correlative conjunction(s), discourse cue(s) (e.g. 'however', 'therefore'), pronoun(s), definite article(s)

Table 3: Features used for determining adverbial positions. We did not find phrases with more than 5 adverbial siblings or more more than 5 non-adverbial siblings. If a phrase did not have 5 adverbial or non-adverbial siblings, NA values were used in the features for those siblings.

Two-stage For our two-stage classification-based approach, we first determine the position of each adverbial in a phrase in relation to its non-adverbial siblings, and then the relative positions of adjacent adverbials. For the first stage we use a classifier. There is one feature vector for each adverbial. It contains features for all non-adverbial siblings in realization order, then for each adverbial sibling in alphabetical order by lexical head, and finally for the target adverbial itself. The label is the position of the target adverbial with respect to the non-adverbial siblings. For our example sentence in Figure 1, the label for "Then" would be 0; for "at the bank" would be 2, and for "on Tuesday" would be 2. If there are n non-adverbial siblings, then there are $n + 1$ possible labels for each feature vector, so the performance of this classifier by chance would be .25 if each adverbial has on average three non-adverbial siblings.

For the second stage we use the same second stage as the baseline approach.

3.2 Method

We use 10-fold cross-validation to compute performance of each approach. For the classifiers, we used the J4 decision tree classifier provided by Weka[1]. We compute correctness for each approach as the percentage of adverbials for which the approach outputs the same position as that found in the original human-produced phrase. (In some cases, multiple positions for the adverbial would be equally acceptable, but we cannot evaluate this automatically.)

[1]We experimented with logistic regression and SVM classifiers; the decision tree classifier gave the highest performance.

3.3 Results

Our classification results are shown in Table 4. The one- and two-stage approaches both significantly outperform baseline. Also, the two-stage approach outperforms the one-stage approach for WSJ.

The decision trees using all features are quite large. We tried dropping feature sets to see if we could get smaller trees without large drops in performance. We found that for all data sets, the models containing only syntactic features perform only about 1% worse for one-stage classification and only about 3% worse for two-stage classification, while in most cases giving much smaller trees (1015 [WSJ] and 972 [SWBD] nodes for the one-stage approach; 1008 [WSJ] and 877 [SWBD] for the two-stage approach). This is somewhat surprising given Costa's arguments about the need for lexical and discourse features; it may be due to errors introduced by approximating discourse features automatically, as well as to data sparsity in the lexical features.

There are only small performance differences between the classifiers for speech and those for text.

4 Realization Experiment

To investigate how a model of adverbial positioning may improve an NLP application, we incorpo-

Approach	Tree size	Classification accuracy	SSA
WSJ			
baseline	n/a	45.98	75.1
one-stage	6519	84.43	82.2
two-stage	1053	86.27	85.1
SWBD			
baseline	n/a	41.48	61.3
one-stage	4486	85.13	74.5
two-stage	3707	85.01	73.1

Table 4: Performance of adverbial position determination

rated our best-performing models into a surface realizer. We automatically extracted a probabilistic lexicalized tree-adjoining grammar from the whole WSJ and SWBD corpora minus our held-out data, using the method described in (Zhong and Stent, 2005). We automatically re-realized all adverbial-containing sentences in our held-out data (10%), after first automatically constructing input using the method described in (Zhong and Stent, 2005).

We compute realization performance using simple string accuracy (SSA)[2]. Realization performance is reported in Table 4. Both classification-based approaches outperform baseline, with the two-stage approach performing best for WSJ with either metric (for SWBD, the classification-based approaches perform similarly).

5 Conclusions and Future Work

In this paper, we tested classification-based approaches to adverbial positioning. We showed that we can achieve good results using syntactic features alone, with small improvements from adding lexical, semantic and sentence-level features. We also showed that use of a model for adverbial positioning leads to improved surface realization. In future work, we plan a human evaluation of our results to see if more features could lead to performance gains.

6 Acknowledgments

This research was partially supported by the NSF under grant no. 0325188.

[2]Although in general we do not find SSA to be a reliable metric for evaluating surface realizers, in this case it is valid because lexical selection is done already; only the positions of adverbials will generally be different.

References

D. Allen and A. Cruttenden. 1974. English sentence adverbials: Their syntax and their intonation in British English. *Lingua*, 34:1–30.

J. Burstein, M. Chodorow, and C. Leacock. 2004. Automated essay evaluation: The Criterion online writing service. *AI Magazine*, 25(3):27–36.

J. Clarke and M. Lapata. 2007. Modelling compression with discourse constraints. In *Proceedings of EMNLP/CoNLL*.

J. Costa. 2004. A multifactorial approach to adverb placement: assumptions, facts, and problems. *Lingua*, 114:711–753.

M. Elhadad, Y. Netzer, R. Barzilay, and K. McKeown. 2001. Ordering circumstantials for multi-document summarization. In *Proceedings of BISFAI*.

Thomas Ernst. 1984. *Towards an integrated theory of adverb position in English*. Ph.D. thesis, Indiana University, Bloomington, Indiana.

H. Haider. 2000. Adverb placement. *Theoretical linguistics*, 26:95–134.

J. Keyser. 1968. Adverbial positions in English. *Language*, 44:357–374.

C. Leacock. 2007. Writing English as a second language: A proofreading tool. In *Proceedings of the Workshop on optimizing the role of language in technology-enhanced learning*.

N. Madnani, D. Zajic, B. Dorr, N. F. Ayan, and J. Lin. 2007. Multiple alternative sentence compressions for automatic text summarization. In *Proceedings of the Document Understanding Conference*.

T. Marciniak and M. Strube. 2004. Classification-based generation using TAG. In *Lecture Notes in Computer Science*, volume 3123/2004. Springer Berlin/Heidelberg.

M. Marcus, B. Santorini, M. Marcinkiewicz, and A. Taylor. 1999. Treebank-3. Available from the Linguistic Data Consortium, Catalog Number LDC99T42.

K. Ogura, S. Shirai, and F. Bond. 1997. English adverb processing in Japanese-to-English machine translation. In *Seventh International Conference on Theoretical and Methodological Issues in Machine Translation*.

E. Ringger, M. Gamon, R. Moore, D. Rojas, M. Smets, and S. Corston-Oliver. 2004. Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proceedings of COLING*.

H. Zhong and A. Stent. 2005. Building surface realizers automatically from corpora using general-purpose tools. *Proceedings of UCNLG*.

H. Zhong and A. Stent. 2008. A corpus-based comparison of models for predicting ordering of prepositional phrases. *In submission*.

Estimating and Exploiting the Entropy of Sense Distributions

Peng Jin
Institute of Computational Linguistics
Peking University
Beijing China
jandp@pku.edu.cn

Diana McCarthy, Rob Koeling and **John Carroll**
University of Sussex
Falmer, East Sussex
BN1 9QJ, UK
{dianam,robk,johnca}@sussex.ac.uk

Abstract

Word sense distributions are usually skewed. Predicting the extent of the skew can help a word sense disambiguation (WSD) system determine whether to consider evidence from the local context or apply the simple yet effective heuristic of using the first (most frequent) sense. In this paper, we propose a method to estimate the entropy of a sense distribution to boost the precision of a first sense heuristic by restricting its application to words with lower entropy. We show on two standard datasets that automatic prediction of entropy can increase the performance of an automatic first sense heuristic.

1 Introduction

Word sense distributions are typically skewed and WSD systems do best when they exploit this tendency. This is usually done by estimating the most frequent sense (MFS) for each word from a training corpus and using that sense as a back-off strategy for a word when there is no convincing evidence from the context. This is known as the MFS heuristic [1] and is very powerful since sense distributions are usually skewed. The heuristic becomes particularly hard to beat for words with highly skewed sense distributions (Yarowsky and Florian, 2002). Although the MFS can be estimated from tagged corpora, there are always cases where there is insufficient data, or where the data is inappropriate, for example because it comes from a very different domain. This has motivated some recent work attempting to estimate the distributions automatically (McCarthy et al., 2004; Lapata and Keller, 2007). This paper examines the case for determining the skew of a word sense distribution by estimating entropy and then using this to increase the precision of an unsupervised first sense heuristic by restricting application to those words where the system can automatically detect that it has the most chance. We use a method based on that proposed by McCarthy et al. (2004) as this approach does not require hand-labelled corpora. The method could easily be adapted to other methods for predicing predominant sense.

[1] It is also referred to as the first sense heuristic in the WSD literature and in this paper.

2 Method

Given a listing of senses from an inventory, the method proposed by McCarthy et al. (2004) provides a prevalence ranking score to produce a MFS heuristic. We make a slight modification to McCarthy et al.'s prevalence score and use it to estimate the probability distribution over the senses of a word. We use the same resources as McCarthy et al. (2004): a distributional similarity thesaurus and a WordNet semantic similarity measure. The thesaurus was produced using the metric described by Lin (1998) with input from the grammatical relation data extracted using the 90 million words of written English from the British National Corpus (BNC) (Leech, 1992) using the RASP parser (Briscoe and Carroll, 2002). The thesaurus consists of entries for each word (w) with the top 50 "nearest neighbours" to w, where the neighbours are words ranked by the distributional similarity that

Proceedings of NAACL HLT 2009: Short Papers, pages 233–236,
Boulder, Colorado, June 2009.

they share with w. The WordNet similarity score is obtained with the **jcn** measure (Jiang and Conrath, 1997) using the WordNet Similarity Package 0.05 (Patwardhan and Pedersen, 2003) and WordNet version 1.6. The **jcn** measure needs word frequency information, which we obtained from the BNC.

2.1 Estimates of Predominance, Probability and Entropy

Following McCarthy et al. (2004), we calculate prevalence of each sense of the word (w) using a weighted sum of the distributional similarity scores of the top 50 neighbours of w. The sense of w that has the highest value is the automatically detected MFS (predominant sense). The weights are determined by the WordNet similarity between the sense in question and the neighbour. We make a modification to the original method by multiplying the weight by the inverse rank of the neighbour from the list of 50 neighbours. This modification magnifies the contribution to each sense depending on the rank of the neighbour while still allowing a neighbour to contribute to all senses that it relates too. We verified the effect of this change compared to the original ranking score by measuring cross-entropy. [2]

Let $N_w = n_1, n_2 \ldots n_k$ denote the ordered set of the top $k = 50$ neighbours of w according to the distributional similarity thesaurus, $senses(w)$ is the set of senses of w and $dss(w, n_j)$ is the distributional similarity score of a word w and its j^{th} neighbour. Let ws_i be a sense of w then $wnss(ws_i, n_j)$ is the maximum WordNet similarity score between ws_i and the WordNet sense of the neighbour (n_j) that maximises this score. The prevalence score is calculated as follows with $\frac{1}{rank_{n_j}}$ being our modification to McCarthy et al.

$$Prevalence\ Score(ws_i) = \sum_{n_j \in N_w} dss(w, n_j) \times \frac{wnss(ws_i, n_j)}{\sum_{ws_{i'} \in senses(w)} wnss(ws_{i'}, n_j)} \times \frac{1}{rank_{n_j}} \quad (1)$$

To turn this score into a probability estimate we sum the scores over all senses of a word and the probability for a sense is the original score divided by this sum:

$$\hat{p}(ws_i) = \frac{prevalence\ score(ws_i)}{\sum_{ws_j \in w} prevalence\ score(ws_j)} \quad (2)$$

To smooth the data, we evenly distribute 1/10 of the smallest prevalence score to all senses with a undefined prevalence score values. Entropy is measured as:

$$H(senses(w)) = - \sum_{ws_i \in senses(w)} p(ws_i) log(p(ws_i))$$

using our estimate ($\hat{p}$) for the probability distribution p over the senses of w.

3 Experiments

We conducted two experiments to evaluate the benefit of using our estimate of entropy to restrict application of the MFS heuristic. The two experiments are conducted on the polysemous nouns in SemCor and the nouns in the SENSEVAL-2 English all words task (we will refer to this as SE2-EAW).

3.1 SemCor

For this experiment we used all the polysemous nouns in Semcor 1.6 (excluding multiwords and proper nouns). We depart slightly from (McCarthy et al., 2004) in including all polysemous nouns whereas they limited the experiment to those with a frequency in SemCor of 3 or more and where there is one sense with a higher frequency than the others. Table 1 shows the precision of finding the predominant sense using equation 1 with respect to different entropy thresholds. At each threshold, the MFS in Semcor provides the upper-bound (UB). The random baseline (RBL) is computed by selecting one of the senses of the target word randomly as the predominant sense. As we hypothesized, precision is higher when the entropy of the sense distribution is lower, which is an encouraging result given that the entropy is automatically estimated. The performance of the random baseline is higher at lower entropy which shows that the task is easier and involves a lower degree of polysemy of the target words. However, the gains over the random baseline are greater at lower entropy levels indicating that the merits of detecting the skew of the distribution cannot all be due to lower polysemy levels.

[2] Our modified version of the score gave a lower cross-entropy with SemCor compared to that in McCarthy et al. The result was highly significant with $p < 0.01$ on the t-test.

H ($\leq$)	precision eq 1	RBL	UB	# tokens
0.5	-	-	-	0
0.9	80.3	50.0	84.8	466
0.95	85.1	50.0	90.9	1360
1	68.5	50.0	87.4	9874
1.5	67.6	42.6	86.9	11287
2	58.0	36.7	79.5	25997
2.5	55.7	34.4	77.6	31599
3.0	50.2	30.6	73.4	41401
4.0	47.6	28.5	70.8	46987
5.0 (all)	47.3	27.3	70.5	47539

Table 1: First sense heuristic on SemCor

Freq $\leq$	P	#tokens
1	45.9	1132
5	50.1	5765
10	50.7	10736
100	49.4	39543
1000(all)	47.3	47539
#senses $\leq$	P	#tokens
2	67.2	10736
5	55.4	31181
8	50.1	41393
12	47.8	46041
30(all)	47.3	47539

Table 2: Precision (P) of equation 1 on SemCor with respect to frequency and polysemy

We also conducted a frequency and polysemy analysis shown in Table 2 to demonstrate that the increase in precision is not all due to frequency or polysemy. This is important, since both frequency and polysemy level (assuming a predefined sense inventory) could be obtained without the need for automatic estimation. As we can see, while precision is higher for lower polysemy, the automatic estimate of entropy can provide a greater increase in precision than polysemy, and frequency does not seem to be strongly correlated with precision.

3.2 SENSEVAL-2 English All Words Dataset

The SE2-EAW task provides a hand-tagged test suite of 5,000 words of running text from three articles from the Penn Treebank II (Palmer et al., 2001). Again, we examine whether precision of the MFS heuristic can be increased by restricting application depending on entropy. We use the same resources as for the SemCor experiment. [3] Table 3 gives the results. The most frequent sense (MFS) from SE2-EAW itself provides the upper-bound (UB). We also compare performance with the Semcor MFS (SC). Performance is close to the Semcor MFS while not relying on any manual tagging. As before, precision increases significantly for words with low estimated entropy, and the gains over the random baseline are higher compared to the gains including all words.

H ($\leq$)	precision eq 1	RBL	SC	UB	# tokens
0.5	-	-	-	-	0
0.9	1	50.0	1	1	7
0.95	94.7	50.0	94.7	1	19
1	69.6	50.0	81.3	94.6	112
1.5	68.0	49.0	81.3	93.8	128
2	69.6	34.7	68.2	87.7	421
2.5	65.0	33.0	65.0	86.5	488
3.0	56.6	27.5	60.8	80.1	687
4.0	52.6	25.6	58.8	79.2	766
5.0 (all)	51.5	25.6	58.5	79.3	769

Table 3: First sense heuristic on SE2-EAW

4 Related Work

There is promising related work on determining the predominant sense for a MFS heuristic (Lapata and Keller, 2007; Mohammad and Hirst, 2006) but our work is the first to use the ranking score to estimate entropy and apply it to determine the confidence in the MFS heuristic. It is likely that these methods would also have increased precision if the ranking scores were used to estimate entropy. We leave such investigations for further work.

Chan and Ng (2005) estimate word sense distributions and demonstrate that sense distribution estimation improves a supervised WSD classifier. They use three sense distribution methods, including that of McCarthy et al. (2004). While the other two methods outperform the McCarthy et al. method,

[3] We also used a tool for mapping from WordNet 1.7 to WordNet 1.6 (Daudé et al., 2000) to map the SE2-EAW noun data (originally distributed with 1.7 sense numbers) to 1.6 sense numbers.

they rely on parallel training data and are not applicable on 9.6% of the test data for which there are no training examples. Our method does not require parallel training data.

Agirre and Martínez (2004) show that sense distribution estimation is very important for both supervised and unsupervised WSD. They acquire tagged examples on a large scale by querying Google with monosemous synonyms of the word senses in question. They show that the method of McCarthy et al. (2004) can be used to produce a better sampling technique than relying on the bias from web data or randomly selecting the same number of examples for each sense. Our work similarly shows that the automatic MFS is an unsupervised alternative to SemCor but our work does not focus on sampling but on an estimation of confidence in an automatic MFS heuristic.

5 Conclusions

We demonstrate that our variation of the McCarthy et al. (2004) method for finding a MFS heuristic can be used for estimating the entropy of a sense distribution which can be exploited to boost precision. Words which are estimated as having lower entropy in general get higher precision. This suggests that automatic estimation of entropy is a good criterion for getting higher precision. This is in agreement with Kilgarriff and Rosenzweig (2000) who demonstrate that entropy is a good measure of the difficulty of WSD tasks, though their measure of entropy was taken from the gold-standard distribution itself.

As future work, we want to compare this approach of estimating entropy with other methods for estimating sense distributions which do not require hand-labelled data or parallel texts. Currently, we disregard local context. We wish to couple the confidence in the MFS with contextual evidence and investigate application on coarse-grained datasets.

Acknowledgements

This work was funded by the China Scholarship Council, the National Grant Fundamental Research 973 Program of China: Grant No. 2004CB318102, the UK EPSRC project EP/C537262 'Ranking Word Senses for Disambiguation', and a UK Royal Society Dorothy Hodgkin Fellowship to the second author.

References

E. Agirre and D. Martínez. 2004. Unsupervised wsd based on automatically retrieved examples: The importance of bias. In *Proceedings of EMNLP-2004*, pages 25–32, Barcelona, Spain.

E. Briscoe and J. Carroll. 2002. Robust accurate statistical annotation of general text. In *Proceedings of LREC-2002*, pages 1499–1504, Las Palmas, Canary Islands, Spain.

Y.S. Chan and H.T. Ng. 2005. Word sense disambiguation with distribution estimation. In *Proceedings of IJCAI 2005*, pages 1010–1015, Edinburgh, Scotland.

J. Daudé, L. Padró, and G. Rigau. 2000. Mapping wordnets using structural information. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, Hong Kong.

J. Jiang and D. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In *International Conference on Research in Computational Linguistics*, Taiwan.

A. Kilgarriff and J. Rosenzweig. 2000. Framework and results for english SENSEVAL. *Computers and the Humanities. Senseval Special Issue*, 34(1–2):15–48.

M. Lapata and F. Keller. 2007. An information retrieval approach to sense ranking. In *Proceedings of NAACL-2007*, pages 348–355, Rochester.

G. Leech. 1992. 100 million words of English: the British National Corpus. *Language Research*, 28(1):1–13.

D. Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of COLING-ACL 98*, Montreal, Canada.

D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. 2004. Finding predominant senses in untagged text. In *Proceedings of ACL-2004*, pages 280–287, Barcelona, Spain.

S. Mohammad and G. Hirst. 2006. Determining word sense dominance using a thesauru s. In *Proceedings of EACL-2006*, pages 121–128, Trento, Italy.

M. Palmer, C. Fellbaum, S. Cotton, L. Delfs, and H. Trang Dang. 2001. English tasks: All-words and verb lexical sample. In *Proceedings of the* SENSEVAL-2 *workshop*, pages 21–24.

S. Patwardhan and T. Pedersen. 2003. The wordnet::similarity package. http://wn-similarity.sourceforge.net/.

D. Yarowsky and R. Florian. 2002. Evaluating sense disambiguation performance across diverse parameter spaces. *Natural Language Engineering*, 8(4):293–310.

Semantic classification with WordNet kernels

Diarmuid Ó Séaghdha
Computer Laboratory
University of Cambridge
United Kingdom
do242@cl.cam.ac.uk

Abstract

This paper presents methods for performing graph-based semantic classification using kernel functions defined on the WordNet lexical hierarchy. These functions are evaluated on the SemEval Task 4 relation classification dataset and their performance is shown to be competitive with that of more complex systems. A number of possible future developments are suggested to illustrate the flexibility of the approach.

1 Introduction

The estimation of semantic similarity between words is one of the longest-established tasks in Natural Language Processing and many approaches to the problem have been proposed. The two dominant lexical similarity paradigms are distributional similarity, which compares words on the basis of their observed co-occurrence behaviour in corpora, and semantic network similarity, which compares words based on their position in a graph such as the WordNet hierarchy. In this paper we consider measures of network similarity for the purpose of supervised classification with kernel methods. The utility of kernel functions related to popular distributional similarity measures has recently been demonstrated by Ó Séaghdha and Copestake (2008); we show here that kernel analogues of WordNet similarity can likewise give good performance on a semantic classification task.

2 Kernels derived from graphs

Kernel-based classifiers such as support vector machines (SVMs) make use of functions called *kernel functions* (or simply *kernels*) to compute the similarity between data points (Shawe-Taylor and Cristianini, 2004). Valid kernels are restricted to the set of *positive semi-definite (psd) functions*, i.e., those that correspond to an inner product in some vector space. Kernel methods have been widely adopted in NLP over the past decade, in part due to the good performance of SVMs on many tasks and in part due to the ability to exploit prior knowledge about a given task through the choice of an appropriate kernel function. In this section we consider kernel functions that use spectral properties of a graph to compute the similarity between its nodes. The theoretical foundations and some machine learning applications of the adopted approach have been developed by Kondor and Lafferty (2002), Smola and Kondor (2003) and Herbster et al. (2008).

Let G be a graph with vertex set $V = v_1, \ldots, v_n$ and edge set $E \subseteq V \times V$. We assume that G is connected and undirected and that all edges have a positive weight $w_{ij} > 0$. Let $\mathbf{A}$ be the symmetric $n \times n$ matrix with entries $A_{ij} = w_{ij}$ if an edge exists between vertices v_i and v_j, and $A_{ij} = 0$ otherwise. Let $\mathbf{D}$ be the diagonal matrix with entries $D_{ii} = \sum_{j \in V} A_{ij}$. The *graph Laplacian* $\mathbf{L}$ is then defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{A} \tag{1}$$

The normalised Laplacian is defined as $\hat{\mathbf{L}} = \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$. Both $\hat{\mathbf{L}}$ and $\mathbf{L}$ are positive semi-definite, but they are typically used as starting points

Proceedings of NAACL HLT 2009: Short Papers, pages 237–240,
Boulder, Colorado, June 2009.

for the derivation of kernels rather than as kernels themselves.

Let $\lambda_1 \geq \cdots \geq \lambda_n$ be the eigenvalues of $\mathbf{L}$ and $u_1, \ldots, u_n$ the corresponding eigenvectors. Note that $u_n = 0$ for all graphs. $\mathbf{L}$ is singular and hence has no well-defined inverse, but its pseudoinverse $\mathbf{L}^+$ is defined as

$$\mathbf{L}^+ = \sum_{i=1}^{n-1} \lambda_i^{-1} u_i u_i^{\mathrm{T}} \quad (2)$$

$\mathbf{L}^+$ is positive definite, and its entries are related to the *resistance distance* between points in an electrical circuit (Herbster et al., 2008) and to the *average commute-time distance*, i.e., the average distance of a random walk from one node to another and back again (Fouss et al., 2007). The similarity measure defined by $\mathbf{L}^+$ hence takes information about the connectivity of the graph into account as well as information about adjacency. An analogous pseudoinverse $\hat{\mathbf{L}}^+$ can be defined for the normalised Laplacian.

A second class of graph-based kernel functions are the *diffusion kernels* introduced by Kondor and Lafferty (2002). The kernel $\mathbf{H}_t$ is defined as $\mathbf{H}_t = e^{-t\hat{\mathbf{L}}}$, or equivalently:

$$\mathbf{H}_t = \sum_{i=1}^{n-1} \exp(-t\hat{\lambda}_i) \hat{u}_i \hat{u}_i^{\mathrm{T}} \quad (3)$$

where $t > 0$, and $\hat{\lambda}_1 \geq \cdots \geq \hat{\lambda}_n$ and $\hat{u}_1, \ldots, \hat{u}_n$ are the eigenvalues and eigenvectors of $\hat{\mathbf{L}}^+$ respectively. $\mathbf{H}_t$ can be interpreted in terms of heat diffusion or the distribution of a lazy random walk emanating from a given point at a time point t.

3 Methodology

3.1 Graph construction

WordNet (Fellbaum, 1998) is a semantic network in which nodes correspond to word senses (or *synsets*) and edges correspond to relations between senses. In this work we restrict ourselves to the noun component of WordNet and use only hyponymy and instance hyponymy relations for graph construction. The version of WordNet used is WordNet 3.0.

To evaluate the utility of the graph-based kernels described in Section 2 for computing lexical similarity, we use the dataset developed for the task on Classifying Semantic Relations Between Nominals at the 2007 SemEval competition (Girju et al., 2007). The dataset comprises candidate example sentences for seven two-argument semantic relations, with 140 training sentences and approximately 80 test sentences for each relation. It is a particularly suitable task for evaluating WordNet kernels, as the candidate relation arguments for each sentence are tagged with their WordNet sense and it has been previously shown that a kernel model based on distributional lexical similarity can attain very good performance (Ó Séaghdha and Copestake, 2008).

3.2 Calculating the WordNet kernels

The noun hierarchy in WordNet 3.0 contains 82,115 senses; computing kernel similarities on a graph of this size raises significant computational issues. The calculation of the Laplacian pseudoinverse is complicated by the fact that while $\mathbf{L}$ and $\hat{\mathbf{L}}$ are very sparse, their pseudoinverses are invariably dense and require very large amounts of memory. To circumvent this problem, we follow Fouss et al. (2007) in computing $\mathbf{L}^+$ and $\hat{\mathbf{L}}^+$ one column at a time through a Cholesky factorisation procedure. Only those columns required for the classification task need be calculated, and the kernel computation for each relation subtask can be performed in a matter of minutes. Calculating the diffusion kernel involves an eigendecomposition of $\hat{\mathbf{L}}$, meaning that computing the kernel exactly is infeasible. The solution used here is to approximate $\mathbf{H}_t$ by using the m smallest components of the spectrum of $\hat{\mathbf{L}}$ when computing (3); from (2) it can be seen that a similar approximation can be made to speed up computation of $\mathbf{L}^+$ and $\hat{\mathbf{L}}^+$.

3.3 Experimental setup

For all kernels and relation datasets, the kernel matrix for each argument position was precomputed and normalised so that every diagonal entry equalled 1. A small number of candidate arguments are not annotated with a WordNet sense or are assigned a non-noun sense; these arguments were assumed to have self-similarity equal to 1 and zero similarity to all other arguments. This does not affect the positive semi-definiteness of the kernel matrices. The per-argument kernel matrices were summed to give the kernel matrix for each relation subtask. The ker-

	Full graph		$m = 500$		$m = 1000$	
Kernel	Acc	F	Acc	F	Acc	F
B	72.1	68.4	-	-	-	-
$\mathbf{L}^+$	73.3	69.4	73.2	70.5	73.6	70.6
$\hat{\mathbf{L}}^+$	72.5	70.0	72.7	70.0	**74.1**	**71.0**
$\mathbf{H}^t$	-	-	68.6	64.7	69.8	65.1

Table 1: Results on SemEval Task 4

nels described in Section 2 were compared to a baseline kernel B. This baseline represents each word as a binary feature vector describing its synset and all its hypernym synsets in the WordNet hierarchy, and calculates the linear kernel between vectors.

All experiments were run using the LIBSVM support vector machine library (Chang and Lin, 2001). For each relation the SVM cost parameter was optimised in the range $(2^{-6}, 2^{-4}, \ldots, 2^{12})$ through cross-validation on the training set. The diffusion kernel parameter t was optimised in the same way, in the range $(10^{-3}, 10^{-2}, \ldots, 10^{3})$.

4 Results

Macro-averaged accuracy and F-score for each kernel are reported in Table 1. There is little difference between the Laplacian and normalised Laplacian pseudoinverses; both achieve better performance than the baseline B. The results also suggest that the reduced-eigenspectrum approximations to $\mathbf{L}^+$ and $\hat{\mathbf{L}}^+$ may bring benefits in terms of performance as well as efficiency via a smoothing effect. The best performance is attained by the approximation to $\hat{\mathbf{L}}^+$ with $m = 1,000$ eigencomponents. The heat kernel $\mathbf{H}^t$ fares less well; the problem here may be that the optimal range for the t parameter has not been identified.

Comparing these results to those of the participants in the 2007 SemEval task, the WordNet-based lexical similarity model fares very well. All versions of $\mathbf{L}^+$ and $\hat{\mathbf{L}}^+$ attain higher accuracy than all but one of 15 systems in the competition and higher F-score than all but three. Even the baseline B ranks above all but the top three systems, suggesting that this too can be a useful model. This is in spite of the fact that all systems which made use of the sense annotations also used a rich variety of other information sources such as features extracted from the sentence context, while the models presented here use only the graph structure of WordNet.[1]

5 Related work

There is a large body of work on using WordNet to compute measures of lexical similarity (Budanitsky and Hirst, 2006). However, many of these measures are not amenable for use as kernel functions as they rely on properties which cannot be expressed as a vector inner product, such as the lowest common subsumer of two vertices. Hughes and Ramage (2007) present a lexical similarity model based on random walks on graphs derived from WordNet; Rao et al. (2008) propose the Laplacian pseudoinverse on such graphs as a lexical similarity measure. Both of these works share aspects of the current paper; however, neither address supervised learning or present an application-oriented evaluation.

Extracting features from WordNet for use in supervised learning is a standard technique (Scott and Matwin, 1999). Siolas and d'Alche-Buc (2000) and Basili et al. (2006) use a measure of lexical similarity from WordNet as an intermediary to smooth bag-of-words kernels on documents. Siolas and d'Alche-Buc use an inverse path-based similarity measure, while Basili et al. use a measure of "conceptual density" that is not proven to be positive semi-definite.

6 Conclusion and future work

The main purpose of this paper has been to demonstrate how kernels that capture spectral aspects of graph structure can be used to compare nodes in a lexical hierarchy and thus provide a kernelised measure of WordNet similarity. As far as we are aware, these measures have not previously been investigated in the context of semantic classification. The resulting WordNet kernels have been evaluated on the SemEval Task 4 dataset and shown to attain a higher level of performance than many more complicated systems that participated in that task.

Two obvious shortcomings of the kernels discussed here are that they are defined on senses rather than words and that they are computed on a

[1] Of course, information about lexical similarity is not sufficient to classify all examples. In particular, the models presented here perform relatively badly on the ORIGIN-ENTITY and THEME-TOOL relations, while scoring better than all SemEval entrants on INSTRUMENT-AGENCY and PRODUCT-PRODUCER.

rather impoverished graph structure (the WordNet hyponym hierarchy is quite tree-like). One of the significant benefits of spectral graph kernels is that they can be computed on arbitrary graphs and are most powerful when graphs have a rich connectivity structure. Some potential future directions that would make greater use of this flexibility include the following:

- A simple extension from sense-kernels to word-kernels involves adding word nodes to the WordNet graph, with an edge linking each word to each of its possible senses. This is similar to the graph construction method of Hughes and Ramage (2007) and Rao et al. (2008). However, preliminary experiments on the SemEval Task 4 dataset indicate that further refinement of this approach may be necessary in order to match the performance of kernels based on distributional lexical similarity (Ó Séaghdha and Copestake, 2008).
- Incorporating other WordNet relations such as meronymy and topicality gives a way of kernelising semantic association or relatedness; one application of this might be in developing supervised methods for spelling correction (Budanitsky and Hirst, 2006).
- A WordNet graph can be augmented with information from other sources, such as links based on corpus-derived similarity. Alternatively, the graph-based kernel functions could be applied to graphs constructed from parsed corpora (Minkov and Cohen, 2008).

References

Roberto Basili, Marco Cammisa, and Alessandro Moschitti. 2006. A semantic kernel to classify texts with very few training examples. *Informatica*, 30(2):163–172.

Alexander Budanitsky and Graeme Hirst. 2006. Evaluating WordNet-based measures of lexical semantic relatedness. *Computational Linguistics*, 32(1):13–47.

Chih-Chung Chang and Chih-Jen Lin. 2001. LIBSVM: A library for support vector machines. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

Christiane Fellbaum, editor. 1998. *WordNet: An Electronic Lexical Database*. MIT Press, Cambridge, MA.

Francois Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. 2007. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 19(3):355–369.

Roxana Girju, Preslav Nakov, Vivi Nastase, Stan Szpakowicz, Peter Turney, and Deniz Yuret. 2007. SemEval-2007 Task 04: Classification of semantic relations between nominals. In *Proceedings of the 4th International Workshop on Semantic Evaluations (SemEval-07)*.

Mark Herbster, Massimiliano Pontil, and Sergio Rojas Galeano. 2008. Fast prediction on a tree. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS-08)*.

Thad Hughes and Daniel Ramage. 2007. Lexical semantic relatedness with random graph walks. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL-07)*.

Risi Imre Kondor and John Lafferty. 2002. Diffusion kernels on graphs and other discrete input spaces. In *Proceedings of the 19th International Conference on Machine Learning (ICML-02)*.

Einat Minkov and William W. Cohen. 2008. Learning graph walk based similarity measures for parsed text. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing (EMNLP-08)*.

Diarmuid Ó Séaghdha and Ann Copestake. 2008. Semantic classification with distributional kernels. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING-08)*.

Delip Rao, David Yarowsky, and Chris Callison-Burch. 2008. Affinity measures based on the graph Laplacian. In *Proceedings of the 3rd TextGraphs Workshop on Graph-based Algorithms for NLP*.

Sam Scott and Stan Matwin. 1999. Feature engineering for text classification. In *Proceedings of the 16th International Conference on Machine Learning (ICML-99)*.

John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press, Cambridge.

Georges Siolas and Florence d'Alche-Buc. 2000. Support vector machines based on a semantic kernel for text categorization. In *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*.

Alexander J. Smola and Risi Kondor. 2003. Kernels and regularization on graphs. In *Proceedings of the the 16th Annual Conference on Learning Theory and 7th Workshop on Kernel Machines (COLT-03)*.

Sentence Boundary Detection and the Problem with the U.S.

Dan Gillick
Computer Science Division
University of California, Berkeley
dgillick@cs.berkeley.edu

Abstract

Sentence Boundary Detection is widely used but often with outdated tools. We discuss what makes it difficult, which features are relevant, and present a fully statistical system, now publicly available, that gives the best known error rate on a standard news corpus: Of some 27,000 examples, our system makes 67 errors, 23 involving the word "U.S."

1 Introduction

Many natural language processing tasks begin by identifying sentences, but due to the semantic ambiguity of the period, the sentence boundary detection (SBD) problem is non-trivial. While reported error rates are low, significant improvement is possible and potentially valuable. For example, since a single error can ruin an automatically generated summary, reducing the error rate from 1% to 0.25% reduces the rate of damaged 10-sentence summaries from 1 in 10 to 1 in 40. Better SBD may improve language models and sentence alignment as well.

SBD has been addressed only a few times in the literature, and each result points to the importance of developing lists of common abbreviations and sentence starters. Further, most practical implementations are not readily available (with one notable exception). Here, we present a fully statistical system that we argue benefits from avoiding manually constructed or tuned lists. We provide a detailed analysis of features, training variations, and errors, all of which are under-explicated in the literature, and discuss the possibility of a more structured classification approach. Our implementation gives the best performance, to our knowledge, reported on a standard Wall Street Journal task; it is open-source and available to the public.

2 Previous Work

We briefly outline the most important existing methods and cite error rates on a standard English data set, sections 03-06 of the Wall Street Journal (WSJ) corpus (Marcus et al., 1993), containing nearly 27,000 examples. Error rates are computed as $(\text{number incorrect}/\text{total ambiguous periods})$. Ambiguous periods are assumed to be those followed by white space or punctuation. Guessing the majority class gives a 26% baseline error rate.

A variety of systems use lists of hand-crafted regular expressions and abbreviations, notably Alembic (Aberdeen et al., 1995), which gives a 0.9% error rate. Such systems are highly specialized to language and genre.

The Satz system (Palmer and Hearst, 1997) achieves a 1.0% error rate using part-of-speech (POS) features as input to a neural net classifier (a decision tree gives similar results), trained on held-out WSJ data. Features were generated using a 5000-word lexicon and a list of 206 abbreviations. Another statistical system, mxTerminator (Reynar and Ratnaparkhi, 1997) employs simpler lexical features of the words to the left and right of the candidate period. Using a maximum entropy classifier trained on nearly 1 million words of additional WSJ data, they report a 1.2% error rate with an automatically generated abbreviation list and special corpus-specific abbreviation features.

There are two notable unsupervised systems. Punkt (Kiss and Strunk, 2006) uses a set of log-likelihood-based heuristics to infer abbreviations and common sentence starters from a large text corpus. Deriving these lists from the WSJ test data gives an error rate of 1.65%. Punkt is easily adaptable but requires a large (unlabeled) in-domain corpus for assembling statistics. An implementation is bundled with NLTK (Loper and Bird, 2002). (Mikheev, 2002) describes a "document-

Proceedings of NAACL HLT 2009: Short Papers, pages 241–244,
Boulder, Colorado, June 2009.

centered" approach to SBD, using a set of heuristics to guess which words correspond to abbreviations and names. Adding carefully tuned lists from an extra news corpus gives an error rate of 0.45%, though this increases to 1.41% without the abbreviation list. Combining with a supervised POS-based system gives the best reported error rate on this task: 0.31%.

Our system is closest in spirit to mxTerminator, and we use the same training and test data in our experiments to aid comparison.

3 Our Approach

Each example takes the general form "$L.\ R$", where L is the context on the left side of the period in question, and R is the context on the right (we use only one word token of context on each side). We are interested in the probability of the binary sentence boundary class s, conditional on its context: $P(s|\text{"}L.\ R\text{"})$. We take a supervised learning approach, extracting features from "$L.\ R$".

Table 1 lists our features and their performance, using a Support Vector Machine (SVM) with a linear kernel[1]. Feature 1 by itself, the token ending with the candidate period, gives surprisingly good performance, and the combination of 1 and 2 outperforms nearly all documented systems. While no published result uses an SVM, we note that a simple Naive Bayes classifier gives an error rate of 1.05% (also considerably better than mxTerminator), suggesting that the choice of classifier alone does not explain the performance gap.

There are a few possible explanations. First, proper tokenization is key. While there is not room to catalog our tokenizer rules, we note that both untokenized text and mismatched train-test tokenization can increase the error rate by a factor of 2.

Second, poor feature choices can hurt classification. In particular, adding a feature that matches a list of abbreviations can increase the error rate; using the list ("Mr.", "Co.") increases the number of errors by up to 25% in our experiments. This is because some abbreviations end sentences often, and others do not. In the test data, 0 of 1866 instances of "Mr." end a sentence, compared to 24 of 86 instances of "Calif." (see Table 2). While there may be meaningful abbreviation subclasses, a feature indicating mere presence is too coarse.

[1]We use SVM Light, with $c = 1$ (Joachims, 1999). Nonlinear kernels did not improve performance in our experiments.

#	Feature Description	Error
1	$L = w_i$	1.88%
2	$R = w_j$	9.36%
3	$len(L) = l$	9.12%
4	$is\_cap(R)$	12.56%
5	$int(log(count(L; \text{no period}))) = c_i$	12.14%
6	$int(log(count(R; \text{is lower})) = c_j$	18.79%
7	$(L = w_i, R = w_j)$	10.01%
8	$(L = w_i, is\_cap(R))$	7.54%
1+2		0.77%
1+2+3+4		0.36%
1+2+3+4+5+6		0.32%
1+2+3+4+5+6+7+8		0.25%

Table 1: All features are binary. SVM classification results shown; Naive Bayes gives 0.35% error rate with all features.

Abbr.	Ends Sentence	Total	Ratio
Inc.	109	683	0.16
Co.	80	566	0.14
Corp.	67	699	0.10
U.S.	45	800	0.06
Calif.	24	86	0.28
Ltd.	23	112	0.21

Table 2: The abbreviations appearing most often as sentence boundaries. These top 6 account for 80% of sentence-ending abbreviations in the test set, though only 5% of all abbreviations.

Adding features 3 and 4 better than cuts the remaining errors in half. These can be seen as a kind of smoothing for sparser token features 1 and 2. Feature 3, the length of the left token, is a reasonable proxy for the abbreviation class (mean abbreviation length is 2.6, compared to 6.1 for non-abbreviation sentence enders). The capitalization of the right token, feature 4, is a proxy for a sentence starter. Every new sentence that starts with a word (as opposed to a number or punctuation) is capitalized, but 70% of words following abbreviations are also, so this feature is mostly valuable in combination.

While we train on nearly 1 million words, most of these are ignored because our features are extracted only near possible sentence boundaries. Consider the fragment "... the U.S. Apparently some ...",

which our system fails to split after "U.S." The word "Apparently" starts only 8 sentences in the training data, but since it usually appears lowercased (89 times in training), its capitalization here is meaningful. Feature 6 encodes this idea, indicating the log count of lowercased appearances of the word right of the candidate period. Similarly, feature 5 gives the log count of occurrences of the token left of the candidate appearing without a final period.

Another way to incorporate all of the training data is to build a model of $P(s|\text{“}L\ R\text{”})$, as is often used in sentence segmentation for speech recognition. Without a period in the conditional, many more negative examples are included. The resulting SVM model is very good at placing periods given input text without them (0.31% error rate), but when limiting the input to examples with ambiguous periods, the error rate is not competitive with our original model (1.45%).

Features 7 and 8 are added to model the nuances of abbreviations at sentence boundaries, helping to reduce errors involving the examples in Table 2.

4 Two Classes or Three?

SBD has always been treated as a binary classification problem, but there are really three classes: sentence boundary only (S); abbreviation only (A); abbreviation at sentence boundary ($A + S$). The label space of the test data, which has all periods annotated, is shown in Figure 1.

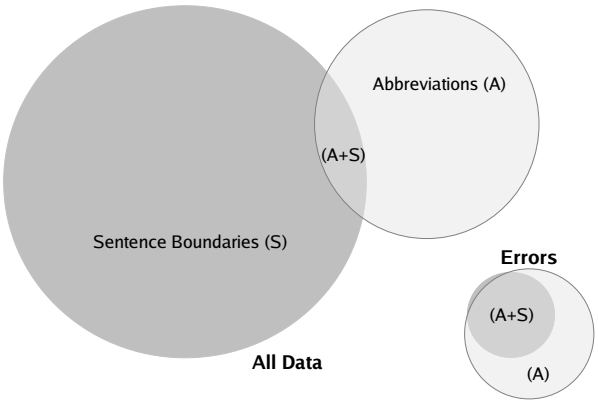


Figure 1: The overlapping label space of the test data: sentence boundaries 74%; abbreviations 26%; intersection 2%. The distribution of errors given by our classifier is shown as well (not to scale with all data).

Relative to the size of the classes, $A + S$ examples are responsible for a disproportionate number of errors, pointing towards the problem with a binary classifier: In the absence of $A + S$ examples, the left context L and the right context R both help distinguish S from A. But $A + S$ cases have L resembling the A class and R resembling the S class.

One possibility is to add a third class, but this does not improve results, probably because we have so few $A + S$ examples. We also tried taking a more structured approach, depicted in Figure 2, but this too fails to improve performance, mostly because the first step, identifying abbreviations without the right context, is too hard. Certainly, the $A + S$ cases are more difficult to identify, but perhaps some better structured approach could reduce the error rate further.

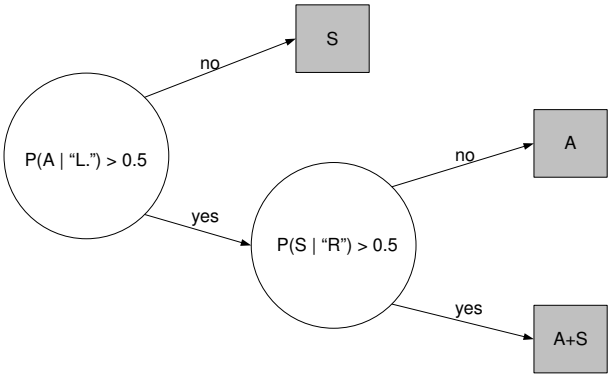


Figure 2: A structured classification approach. The left context is used to separate S examples first, then those remaining are classified as either A or $A + S$ using the right context.

5 Training Data

One common objection to supervised SBD systems is an observation in (Reynar and Ratnaparkhi, 1997), that training data and test data must be a good match, limiting the applicability of a model trained from a specific genre. Table 3 shows respectable error rates for two quite different test sets: The Brown corpus includes 500 documents, distributed across 15 genres roughly representative of all published English; The Complete Works of Edgar Allen Poe includes an introduction, prose, and poetry.

A second issue is a lack of labeled data, especially in languages besides English. Table 4 shows that results can be quite good without extensive labeled resources, and they are likely to continue to improve if additional resources were available. At the least, (Kiss and Strunk, 2006) have labeled over

Corpus	Examples	in S	SVM Err	NB Err
WSJ	26977	74%	0.25%	0.35%
Brown	53688	91%	0.36%	0.45%
Poe	11249	95%	0.52%	0.44%

Table 3: SVM and Naive Bayes classification error rates on different corpora using a model trained from a disjoint WSJ data set.

10000 sentences in each of 11 languages, though we have not experimented with this data.

Corpus	5	50	500	5000	42317
WSJ	7.26%	3.57%	1.36%	0.52%	0.25%
Brown	5.65%	4.46%	1.65%	0.74%	0.36%
Poe	4.01%	2.68%	2.22%	0.98%	0.52%

Table 4: SVM error rates on the test corpora, using models built from different numbers of training sentences.

We also tried to improve results using a standard bootstrapping method. Our WSJ-trained model was used to annotate 100 million words of New York Times data from the AQUAINT corpus, and we included high-confidence examples in a new training set. This did not degrade test error, nor did it improve it.

6 Errors

Our system makes 67 errors out of 26977 examples on the WSJ test set; a representative few are shown in Table 5. 34% of the errors involve the word "U.S." which distinguishes itself as the most difficult of tokens to classify: Not only does it appear frequently as a sentence boundary, but even when it does not, the next word is often capitalized ("U.S. Government"; "U.S. Commission"), further confusing the classifier. In fact, abbreviations for places, including "U.K.", "N.Y.", "Pa." constitute 46% of all errors for the same reason. Most of the remaining errors involve abbreviations like those in Table 2, and all are quite difficult for a human to resolve without more context. Designing features to exploit additional context might help, but could require parsing.

7 Conclusion

We have described a simple yet powerful method for SBD. While we have not tested models in languages other than English, we are providing the code and our models, complete with tokenization, available at *http://code.google.com/p/splitta*. Future work includes further experiments with structured classification to treat the three classes appropriately.

Context	Label	$P(S)$
... the U.S. Amoco already ...	$A+S$	0.45
... the U.K. Panel on ...	A	0.57
... the U.S. Prudential Insurance ...	$A+S$	0.44
... Telephone Corp. President Haruo ...	A	0.73
... Wright Jr. Room ...	A	0.67
... 6 p.m. Travelers who ...	$A+S$	0.44

Table 5: Sample errors with the probability of being in the S class assigned by the SVM.

Acknowledgments

Thanks to Benoit Favre, Dilek Hakkani-Tür, Kofi Boakye, Marcel Paret, James Jacobus, and Larry Gillick for helpful discussions.

References

J. Aberdeen, J. Burger, D. Day, L. Hirschman, P. Robinson, and M. Vilain. 1995. MITRE: description of the Alembic system used for MUC-6. In *Proceedings of the 6th conference on Message understanding*, pages 141–155. Association for Computational Linguistics Morristown, NJ, USA.

T. Joachims. 1999. Making large-scale support vector machine learning practical, Advances in kernel methods: support vector learning.

T. Kiss and J. Strunk. 2006. Unsupervised Multilingual Sentence Boundary Detection. *Computational Linguistics*, 32(4):485–525.

E. Loper and S. Bird. 2002. NLTK: The Natural Language Toolkit. In *Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*, pages 62–69.

M.P. Marcus, M.A. Marcinkiewicz, and B. Santorini. 1993. Building a large annotated corpus of English: the penn treebank. *Computational Linguistics*, 19(2):313–330.

A. Mikheev. 2002. Periods, Capitalized Words, etc. *Computational Linguistics*, 28(3):289–318.

D.D. Palmer and M.A. Hearst. 1997. Adaptive Multilingual Sentence Boundary Disambiguation. *Computational Linguistics*, 23(2):241–267.

J.C. Reynar and A. Ratnaparkhi. 1997. A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 16–19.

Quadratic Features and Deep Architectures for Chunking

Joseph Turian and **James Bergstra** and **Yoshua Bengio**
Dept. IRO, Université de Montréal

Abstract

We experiment with several chunking models. Deeper architectures achieve better generalization. Quadratic filters, a simplification of a theoretical model of V1 complex cells, reliably increase accuracy. In fact, logistic regression with quadratic filters outperforms a standard single hidden layer neural network. Adding quadratic filters to logistic regression is almost as effective as feature engineering. Despite predicting each output label independently, our model is competitive with ones that use previous decisions.

1 Introduction

There are three general approaches to improving chunking performance: engineer better features, improve inference, and improve the model.

Manual feature engineering is a common direction. One technique is to take primitive features and manually compound them. This technique is common, and most NLP systems use n-gram based features (Carreras and Màrquez, 2003; Ando and Zhang, 2005, for example). Another approach is linguistically motivated feature engineering, e.g. Charniak and Johnson (2005).

Other works have looked in the direction of improving inference. Rather than predicting each decision independently, previous decisions can be included in the inference process. In this work, we use the simplest approach of modeling each decision independently.

The third direction is by using a better model. If modeling capacity can be added without introducing too many extra degrees of freedom, generalization could be improved. One approach for compactly increasing capacity is to automatically induce intermediate features through the composition of non-linearities, for example SVMs with a non-linear kernel (Kudo and Matsumoto, 2001), inducing compound features in a CRF (McCallum, 2003), neural networks (Henderson, 2004; Bengio and LeCun, 2007), and boosting decision trees (Turian and Melamed, 2006). Recently, Bergstra et al. (2009) showed that capacity can be increased by adding quadratic filters, leading to improved generalization on vision tasks. This work examines how well quadratic filters work for an NLP task. Compared to manual feature engineering, improved models are appealing because they are less task-specific.

We experiment on the task of chunking (Sang and Buchholz, 2000), a syntactic sequence labeling task.

2 Sequence labeling

Besides Collobert and Weston (2008), previous work on sequence labeling usually use previous decisions in predicting output labels. Here we do not take advantage of the dependency between successive output labels. Our approach predicts each output label independently of the others. This allows us to ignore inference during training: The model maximizes the conditional likelihood of each output label independent of the output label of other tokens.

We use a sliding window approach. The output label for a particular focus token x_i is predicted based upon $\bar{k}$ tokens before and after x_i. The entire window is of size $k = 2 \cdot \bar{k} + 1$. Nearly all work on sequence labeling uses a sliding window approach (Kudo and Matsumoto, 2001; Zhang et al., 2002;

Proceedings of NAACL HLT 2009: Short Papers, pages 245–248,
Boulder, Colorado, June 2009.

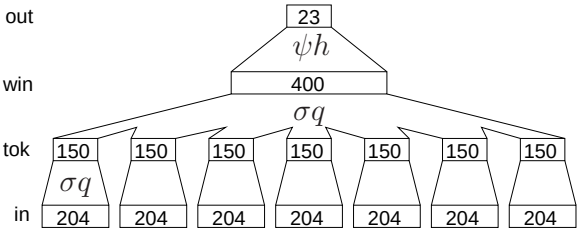


Figure 1: Illustration of our baseline I-T-W-O model (see Secs. 4 and 5.1). The input layer comprises seven tokens with 204 dimensions each. Each token is passed through a shared 150-dimensional token feature extractor. These $7 \cdot 150$ features are concatenated and 400 features are extracted from them in the window layer. These 400 features are the input to the final 23-class output prediction. Feature extractors σq and ψh are described in Section 3.

Carreras and Màrquez, 2003; Ando and Zhang, 2005, for example). We assume that each token x can be transformed into a real-valued feature vector $\phi(x)$ with l entries. The feature function will be described in Section 4.

A standard approach is as follows: We first concatenate the features of k tokens into one vector $[\phi(x_{i-\bar{k}}), \ldots, \phi(x_{i+\bar{k}})]$ of length $k \cdot l$ entries. We can then pass $[\phi(x_{i-\bar{k}}), \ldots, \phi(x_{i+\bar{k}})]$ to a feature extractor over the entire window followed by an output log-linear layer.

Convolutional architectures can help when there is a position-invariant aspect to the input. In machine vision, parameters related to one part of the image are sometimes restricted to be equal to parameters related to another part (LeCun et al., 1998). A convolutional approach to sequence labeling is as follows: At the lowest layer we extract features from individual tokens using a shared feature extractor. These higher-level individual token features are then concatenated, and are passed to a feature extractor over the entire window.

In our baseline approach, we apply one convolutional layer of feature extraction to each token (one *token layer*), followed by a concatenation, followed by one layer of feature extraction over the entire window (one *window layer*), followed by a 23-D output prediction using multiclass logistic regression. We abbreviate this architecture as I-T-W-O (input→token→window→output). See Figure 1 for an illustration of this architecture.

3 Quadratic feature extractors

The most common feature extractor in the literature is a linear filter h followed by a non-linear squashing (activation) function σ:

$$f(x) = \sigma(h(x)), \quad h(x) = \boldsymbol{b} + \boldsymbol{W}x. \tag{1}$$

In our experiments, we use the softsign squashing function $\sigma(z) = z/(1 + |z|)$. n-class logistic regression predicts $\psi(h(x))$, where softmax $\psi_i(z) = \exp(z_i)/\sum_k \exp(z_k)$. Rust et al. (2005) argues that complex cells in the V1 area of visual cortex are not well explained by Eq. 1, but are instead better explained by a model that includes quadratic interactions between regions of the receptive field. Bergstra et al. (2009) approximate the model of Rust et al. (2005) with a simpler model of the form given in Eq. 2.[†] In this model, the pre-squash transformation q includes J quadratic filters:

$$f(x) = \sigma(q(x)), \quad q(x) = \left(\boldsymbol{b} + \boldsymbol{W}x + \sqrt{\sum_{j=1}^{J} (\boldsymbol{V}_j x)^2} \right) \tag{2}$$

where $\boldsymbol{b}$, $\boldsymbol{W}$, and $\boldsymbol{V}_1 \ldots \boldsymbol{V}_J$ are tunable parameters.

In the vision experiments of Bergstra et al. (2009), using quadratic filters improved the generalization of the trained architecture. We were interested to see if the increased capacity would also be beneficial in language tasks. For our logistic regression (I-O) experiments, the architecture is specifically I–ψq→O, i.e. output O is the softmax ψ applied to the quadratic transform q of the input I. Like Bergstra et al. (2009), in architectures with hidden layers, we apply the quadratic transform q in all layers *except* the final layer, which uses linear transform h. For example, I-T-W-O is specifically I–σq→T–σq→W–ψh→O, as shown in Figure 1. Future work will explore if generalization is improved by using q in the final layer.

4 Features

Here is a detailed description of the types of features we use, with number of dimensions:

• **embeddings**. We map each word to a real-valued 50-dimensional embedding. These embeddings were obtained by Collobert and Weston (2008), and

[†] Bergstra et al. (2009) do not use a sqrt in Eq. 2. We found that sqrt improves optimization and gives better generalization.

were induced based upon a purely unsupervised training strategy over the 631 million words in the English Wikipedia.

• **POS-tag**. Part-of-speech tags were assigned automatically, and are part of the CoNLL data. 45 dim.

• **label frequencies**. Frequency of each label assigned to this word in the training and validation data. From Ando and Zhang (2005). 23 dim.

• **type(first character)**. The type of the first character of the word. type(x) = 'A' if x is a capital letter, 'a' if x is a lowercase letter, 'n' if x is a digit, and x otherwise. From Collins (2002). 20 dim.

• **word length**. The length of the word. 20 dim.

• **compressed word type**. We convert each character of the word into its type. We then remove any repeated consecutive type. For example, "Label-making" ⇒ "Aa-a". From Collins (2002). 46 dim.

The last three feature types are based upon orthographic information. There is a combined total of 204 features per token.

5 Experiments

We follow the conditions in the CoNLL-2000 shared task (Sang and Buchholz, 2000). Of the 8936 training sentences, we used 1000 randomly sampled sentences (23615 words) for validation.

5.1 Training details

The optimization criterion used during training is the maximization of the sum (over word positions) of the per-token log-likelihood of the correct decision. Stochastic gradient descent is performed using a fixed learning rate η and early stopping. Gradients are estimated using a minibatch of 8 examples. We found that a learning rate of 0.01, 0.0032, or 0.001 was most effective.

In all our experiments we use a window size of 7 tokens. In preliminary experiments, smaller windows yielded poorer results, and larger ones were no better. Layer sizes of extracted features were chosen to optimize validation F1.

5.2 Results

We report chunk F-measure (F1). In some tables we also report Acc, the per-token label accuracy, *post*-Viterbi decoding.

Figure 2 shows that using quadratic filters reliably improves generalization on all architectures. For the I-T-W-O architecture, quadratic filters increase validation F1 by an absolute 0.31. Most surprisingly, logistic regression with 16 filters achieves F1=93.94, which outperforms the 93.83 of a standard (0 filter) single hidden layer neural network.

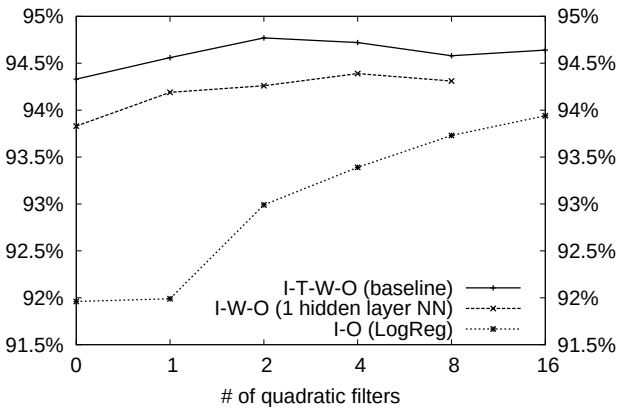


Figure 2: Validation F1 (y-axis) as we vary the number of quadratic filters (x-axis), over different model architectures. Both architecture depth and quadratic filters improve validation F1.

Architecture	#qf	Acc	F1
I-O	16	96.45	93.94
I-W(400)-O	4	96.66	94.39
I-T(150)-W(566)-O	2	96.85	94.77
I-T(150)-W(310)-W(310)-O	4	96.87	94.82

Table 1: Architecture experiments on validation data. The first column describes the layers in the architecture. (The architecture in Figure 1 is I-T(150)-W(400)-O.) The second column gives the number of quadratic filters. For each architecture, the layer sizes and number of quadratic filters are chosen to maximize validation F1. Deeper architectures achieve higher F1 scores.

With embeddings as the only features, logreg with 0 filters achieves F1=85.36. By adding all features, we can raise the F1 to 91.96. Alternately, by adding 16 filters, we can raise the F1 to 91.60. In other words, adding filters is nearly as effective as our manual feature engineering.

Table 1 shows the result of varying the overall architecture. Deeper architectures achieve higher F1 scores. Table 2 compares the model as we lesion off different features. POS tags and the embeddings were the most important features.

We applied our best model overall (I-T-W-W-O in Table 1) to the test data. Results are shown in

Feature set	Acc	F1
default	96.81	94.69
no orthographic features	96.84	94.62
no label frequencies	96.77	94.58
no POS tags	96.60	94.22
no embeddings	96.40	93.97
only embeddings	96.18	93.53

Table 2: Results on validation of varying the feature set, for the architecture in Figure 1 with 4 quadratic filters.

	NP F1	Prc	Rcl	F1
AZ05	94.70	94.57	94.20	94.39
KM01	94.39	93.89	93.92	93.91
I-T-W-W-O	94.44	93.72	93.91	93.81
CM03	94.41	94.19	93.29	93.74
SP03	94.38	-	-	-
Mc03	93.96	-	-	-
AZ05-	-	93.83	93.37	93.60
ZDJ02	93.89	93.54	93.60	93.57

Table 3: Test set results for Ando and Zhang (2005), Kudo and Matsumoto (2001), our I-T-W-W-O model, Carreras and Màrquez (2003), Sha and Pereira (2003), McCallum (2003), Zhang et al. (2002), and our best I-O model. AZ05- is Ando and Zhang (2005) using purely supervised training, not semi-supervised training. Scores are noun phrase F1, and overall chunk precision, recall, and F1.

Table 3. We are unable to compare to Collobert and Weston (2008) because they use a different training and test set. Our model predicts all labels in the sequence independently. All other works in Table 3 use previous decisions when making the current label decision. Our approach is nonetheless competitive with approaches that use this extra information.

6 Conclusions

Many NLP approaches underfit important linguistic phenomena. We experimented with new techniques for increasing chunker model capacity: adding depth (automatically inducing intermediate features through the composition of non-linearities), and including quadratic filters. Higher accuracy was achieved by deeper architectures, i.e. ones with more intermediate layers of automatically tuned feature extractors. Although they are a simplification of a theoretical model of V1 complex cells, quadratic filters reliably improved generalization in all architectures. Most surprisingly, logistic regression with quadratic filters outperformed a single hidden layer neural network without. Also, with logistic regression, adding quadratic filters was almost as effective as manual feature engineering. Despite predicting each output label independently, our model is competitive with ones that use previous decisions.

Acknowledgments

Thank you to Ronan Collobert, Léon Bottou, and NEC Labs for access to their word embeddings, and to NSERC and MITACS for financial support.

References

R. Ando and T. Zhang. A high-performance semi-supervised learning method for text chunking. In *ACL*, 2005.

Y. Bengio and Y. LeCun. Scaling learning algorithms towards AI. In *Large Scale Kernel Machines*. 2007.

J. Bergstra, G. Desjardins, P. Lamblin, and Y. Bengio. Quadratic polynomials learn better image features. TR 1337, DIRO, Université de Montréal, 2009.

X. Carreras and L. Màrquez. Phrase recognition by filtering and ranking with perceptrons. In *RANLP*, 2003.

E. Charniak and M. Johnson. Coarse-to-fine *n*-best parsing and MaxEnt discriminative reranking. In *ACL*, 2005.

M. Collins. Ranking algorithms for named entity extraction: Boosting and the voted perceptron. In *ACL*, 2002.

R. Collobert and J. Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *ICML*, 2008.

J. Henderson. Discriminative training of a neural network statistical parser. In *ACL*, 2004.

T. Kudo and Y. Matsumoto. Chunking with support vector machines. In *NAACL*, 2001.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient based learning applied to document recognition. *IEEE*, 86(11):2278–2324, November 1998.

A. McCallum. Efficiently inducing features of conditional random fields. In *UAI*, 2003.

N. Rust, O. Schwartz, J. A. Movshon, and E. Simoncelli. Spatiotemporal elements of macaque V1 receptive fields. *Neuron*, 46(6):945–956, 2005.

E. T. Sang and S. Buchholz. Introduction to the CoNLL-2000 shared task: Chunking. In *CoNLL*, 2000.

F. Sha and F. C. N. Pereira. Shallow parsing with conditional random fields. In *HLT-NAACL*, 2003.

J. Turian and I. D. Melamed. Advances in discriminative parsing. In *ACL*, 2006.

T. Zhang, F. Damerau, and D. Johnson. Text chunking based on a generalization of Winnow. *JMLR*, 2, 2002.

Active Zipfian Sampling for Statistical Parser Training*

Onur Çobanoğlu
Department of Computer Science
Sennott Square
University of Pittsburgh
Pittsburgh, PA 15260, USA
onc3@pitt.edu

Abstract

Active learning has proven to be a successful strategy in quick development of corpora to be used in training of statistical natural language parsers. A vast majority of studies in this field has focused on estimating informativeness of samples; however, representativeness of samples is another important criterion to be considered in active learning. We present a novel metric for estimating representativeness of sentences, based on a modification of Zipf's *Principle of Least Effort*. Experiments on WSJ corpus with a wide-coverage parser show that our method performs always at least as good as and generally significantly better than alternative representativeness-based methods.

1 Introduction

Wide coverage statistical parsers (Collins, 1997; Charniak, 2000) have proven to require large amounts of manually annotated data for training to achieve substantial performance. However, building such large annotated corpora is very expensive in terms of human effort, time and cost (Marcus et al., 1993). Several alternatives of the standard supervised learning setting have been proposed to reduce the annotation costs, one of which is active learning. Active learning setting allows the learner to select its own samples to be labeled and added to the training data iteratively. The motive behind active learning is that if the learner may select highly informative samples, it can eliminate the redundancy generally found in random data; however, informative samples can be very untypical (Tang et al., 2002). Unlike random sampling, active learning has no guarantee of selecting *representative* samples and untypical training samples are expected to degrade test performance of a classifier.

To get around this problem, several methods of estimating representativeness of a sample have been introduced. In this study, we propose a novel representativeness estimator for a sentence, which is based on a modification of Zipf's *Principle of Least Effort* (Zipf, 1949), theoretically sound and empirically validated on Brown corpus (Francis and Kučera, 1967). Experiments conducted with a wide coverage CCG parser (Clark and Curran, 2004; Clark and Curran, 2007) on CCGbank (Hockenmaier and Steedman, 2005) show that using our estimator as a representativeness metric never performs worse than and generally outperforms length balanced sampling (Becker and Osborne, 2005), which is another representativeness based active learning method, and pure informativeness based active learning.

2 Related Work

In selective sampling setting, there are three criteria to be considered while choosing a sample to add to the training data (Dan, 2004; Tang et al., 2002): *Informativeness* (what will the expected contribution of this sample to the current model be?), *representativeness* (what is the estimated probability of seeing this sample in the target population?) and *diver-*

*Vast majority of this work was done while the author was a graduate student in Middle East Technical University, under the funding from TÜBİTAK-BİDEB through 2210 National Scholarship Programme for MSc Students.

Proceedings of NAACL HLT 2009: Short Papers, pages 249–252,
Boulder, Colorado, June 2009.

sity (how different are the samples in a batch from each other?). The last criterion applies only to the batch-mode setting, in which the training data is incremented by multiple samples at each step for practical purposes.

Most of the active learning research in statistical parser training domain has focused on informativeness measures developed for both single and multi-learner settings. The informativeness measures for single-learners that have exhibited significant performance in well known experimental domains are as follow: Selecting the sentences unparsable by the current model (and if the batch does not get filled, using a secondary method) (Thompson et al., 1999); selecting the sentences with the highest *tree entropy*, i.e. the Shannon entropy of parses the probabilistic parser assigns to the sentence (Hwa, 2004); selecting the sentences having *lowest best probabilities*, where *best probability* is the conditional probability of the most probable parse, given the sentence and the current model (Osborne and Baldridge, 2004); primarily selecting the sentences that are expected to include events observed with low frequency so far with the help of bagging and filling the rest of the batch according to tree entropy, which is named as *two-stage active learning* by Becker and Osborne (2005). Proposed informativeness measures for multiple learners and *ensemble* learners can be found in (Baldridge and Osborne, 2003; Osborne and Baldridge, 2004; Becker and Osborne, 2005; Baldridge and Osborne, 2008).

As for representativeness measures, Tang et. al. (2002) proposed using *sample density*, i.e. the inverse of the average distance of the sample to the other samples in the pool, according to some distance metric. Becker and Osborne (2005) introduced *length balanced sampling*, in which the length histogram of the batch is kept equal to the length histogram of a random sample of batch size drawn from the pool.

3 Description Of The Work

We introduce a novel representativeness measure for statistical parser training domain. Our measure is a function proposed in (Sigurd et al., 2004), which estimates the relative frequencies of sentence lengths in a natural language. Sigurd et. al. (2004) claimed that the longer a sentence is, the less likely it will be uttered; in accordance with Zipf's Principle of Least Effort (Zipf, 1935). However, too short sentences will appear infrequently as well, since the number of different statements that may be expressed using relatively fewer words is relatively smaller. Authors conjectured that there is a clash of expressivity and effort over the frequency of sentence length, which effort eventually wins. They formulated this behavior with a Gamma distribution estimating the relative frequencies of sentence lengths. Authors conducted a parameter fit study for English using Brown corpus (Francis and Kučera, 1967) and reported that the formula $f(L) = 1.1 \times L^1 \times 0.90^L$, where L is the sentence length, fits to the observations with very high correlation.

We propose using this fitted formula (named $f_{zipf-eng}$ from now on) as the measure of representativeness of a sentence. This metric has several nice features: It is model-independent (so it is not affected from modeling errors), is both theoretically sound and empirically validated, can be used in other NLP domains and is a numerical metric, providing flexibility in combining it with informativeness (and diversity) measures.

4 Experiments

4.1 Experimental Setup

We conducted experiments on CCGbank corpus (Hockenmaier and Steedman, 2005) with the wide coverage CCG parser of Clark and Curran (2004; 2007)[1]. C&C parser was fast enough to enable us to use the whole available training data pool for sample selection in experiments, but not for training (since training C&C parser is not that fast). Among the models implemented in the parser, the normal-form model is used. We used the default settings of the C&C parser distribution for fair evaluation. WSJ Sections 02-21 (39604 sentences) are used for training and WSJ Section 23 (2407 sentences) is used for testing. Following (Clark and Curran, 2007), we evaluated the parser performance using the labeled f-score of the predicate-argument dependencies produced by the parser.

[1]Following (Baldridge and Osborne, 2004), we claim that the performances of AL with C&C parser and other state-of-the-art wide coverage parsers will be similar

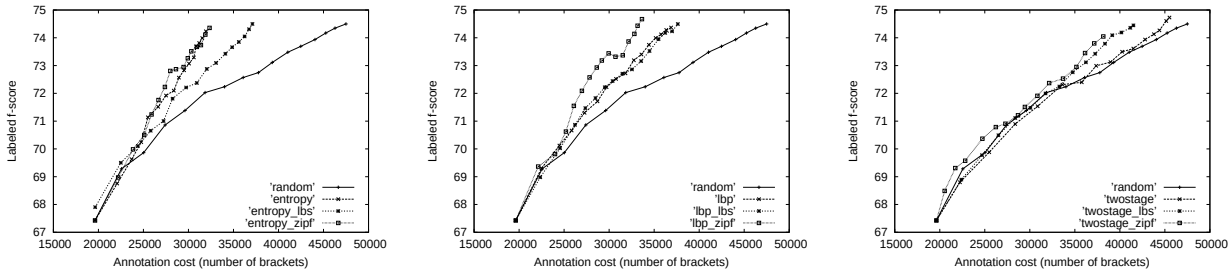


Figure 1: Comparative performances of different representativeness measures. The informativeness measure used is tree entropy in the leftmost graph, lowest best probability in the central graph and two-stage AL in the rightmost graph. The line with the tag 'random' always shows the random sampling baseline.

	none	**lbs**	**zipf**	**random**
entropy	30.99%(74.24%)	20.63%(74.31%)	30.11%(74.36%)	N/A (74.35%)
lbp	22.34%(74.37%)	20.78%(74.49%)	30.19%(74.43%)	N/A(74.50%)
unparsed/entropy	19.98%(74.32%)	19.34%(74.43%)	26.27%(74.38%)	N/A(74.35%)
twostage	2.83%(73.94%)	11.13%(74.09%)	13.38%(74.05%)	N/A(73.94%)

Table 1: PRUD values of different AL schemes. The row includes the informativeness measure and the column includes the representativeness measure used. The column with the label **random** always includes the results for random sampling. The numbers in parentheses are the labeled f-score values reached by the schemes.

For each active learning scheme and random sampling, the size of the seed training set is 500 sentences, the batch size is 100 sentences and iteration stops after reaching 2000 sentences.[2] For statistical significance, each experiment is replicated 5 times. We evaluate the active learning performance in terms of *Percentage Reduction in Utilized Data*, i.e. how many percents less data is used by AL compared to random sampling, in order to reach a certain performance score. Amount of used data is measured with the number of brackets in the data. In CCGbank, a bracket always corresponds to a parse decision, so it is a reasonable approximation of the amount of annotator effort.

Our measure is compared to length balanced sampling and using no representativeness measures. Since there is not a trivial distance metric between CCG parses and we do not know a proposed one, we could not test it against sample density method. We limited the informativeness measures to be tested to the four single-learner measures we mentioned in Section 2. Multi-learner and ensemble methods are excluded, since the success of such methods relies heavily on the diversity of the available models (Baldridge and Osborne, 2004; Baldridge and Osborne, 2008). The models in C&C parser are not diverse enough and we left crafting such diverse models to future work.

We combined $f_{zipf-eng}$ with the informativeness measures as follow: With tree entropy, sentences with the highest $f_{zipf-eng}(s) \times f_{nte}(s, G)$ (named $f_{zipf-entropy}(s, G)$) values are selected. $f_{nte}(s, G)$ is the tree entropy of the sentence s under the current model G, normalized by the binary logarithm of the number of parses, following (Hwa, 2004). With lowest best probability, sentences with the highest $f_{zipf-eng}(s) \times (1 - f_{bp}(s, G))$ values are selected, where f_{bp} is the best probability function (see Section 2). With unparsed/entropy, we primarily chose the unparsable sentences having highest $f_{zipf-eng}(s)$ values and filled the rest of the batch according to $f_{zipf-entropy}$. With two-stage active learning, we primarily chose sentences that can be parsed by the full parser but not the bagged parser and have the highest $f_{zipf-eng}(s)$ values, we secondarily chose sentences that cannot be parsed by both parsers and have the highest $f_{zipf-eng}(s)$ values, the third priority is given to sentences having highest

[2]These values apply to the training of the parser and the CCG supertagger. POS-tagger is trained with the whole available pool of 39604 sentences due to sparse data problem.

$f_{zipf-entropy}$ values.[3] Combining length balanced sampling with all of these informativeness measures is straightforward. For statistical significance, a different random sample is used for length histogram in each replication of experiment.

4.2 Results

Results can be seen in Figure 1 and Table 1. Due to lack of space and similarity of the graphs of unparsed/entropy and LBP, we excluded the graph of unparsed/entropy (but its results are included in Table 1). Since observation points in different lines do not fall on the exactly same performance level (for exact PRUD measurement), we took the points on as closest f-score levels as possible. With tree entropy, Zipfian sampling performs almost as good as pure informativeness based AL and with two-stage AL, length balanced sampling performs almost as good as Zipfian sampling. In all other comparisons, Zipfian sampling outperforms its alternatives substantially.

5 Conclusion and Future Work

We introduced a representativeness measure for active learning in statistical parser training domain, based on an empirical sentence length frequency model of English. Experiments on a wide coverage CCG parser show that this measure outperforms the alternative measures most of the time and never hinders. Our study can be extended via further experimentation with the methods we excluded in Section 4.1, with other parsers, with other languages and with other Zipfian cues of language (e.g. Zipf's law on word frequencies (Zipf, 1949)).

Acknowledgments

We specially thank to Jason Baldridge, Cem Bozşahin, Ruken Çakıcı, Rebecca Hwa, Miles Osborne and anonymous reviewers for their invaluable support, advices and feedback.

References

Jason Baldridge and Miles Osborne. 2003. Active learning for HPSG parse selection. In *Proceedings of CoNLL*.

Jason Baldridge and Miles Osborne. 2004. Active learning and the total cost of annotation. In *Proceedings of EMNLP*.

Jason Baldridge and Miles Osborne. 2008. Active learning and logarithmic opinion pools for HPSG parse selection. In *Natural Language Engineering*, volume 14, pages 199–222. Cambridge, UK.

Markus Becker and Miles Osborne. 2005. A two-stage method for active learning of statistical grammars. In *Proceedings of IJCAI*.

Eugene Charniak. 2000. A maximum-entropy-inspired parser. In *Proceedings of NAACL*.

Stephen Clark and James R. Curran. 2004. Parsing the WSJ using CCG and log-linear models. In *Proceedings of ACL*.

Stephen Clark and James R. Curran. 2007. Wide-coverage efficient statistical parsing with CCG and log-linear models. *Computational Linguistics*, 33(4):493–552.

Michael Collins. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of ACL*.

Shen Dan. 2004. Multi-criteria based active learning for named entity recognition. Master's thesis, National University of Singapore.

W. Nelson Francis and Henry Kučera. 1967. *Computational Analysis of Present-day American English*. Brown University Press, Providence, RI.

Julia Hockenmaier and Mark Steedman. 2005. *CCGbank*. Linguistic Data Consortium, Philadelphia.

Rebecca Hwa. 2004. Sample selection for statistical parsing. *Computational Linguistics*, 30(3):253–276.

Mitchell P. Marcus, Mary A. Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of English:The Penn Treebank. *Computational Linguistics*, 19(2):313–330.

Miles Osborne and Jason Baldridge. 2004. Ensemble-based active learning for parse selection. In *Proceedings of HLT-NAACL*.

Bengt Sigurd, Mats Eeg-Olofsson, and Joost van Weijer. 2004. Word length, sentence length and frequency - Zipf revisited. *Studia Linguistica*, 58(1):37–52.

Min Tang, Xiaoqiang Luo, and Salim Roukos. 2002. Active learning for statistical natural language parsing. In *Proceedings of ACL*.

Cynthia A. Thompson, Mary E. Califf, and Raymond J. Mooney. 1999. Active learning for natural language parsing and information extraction. In *Proceedings of ICML*.

George K. Zipf. 1935. *The Psychobiology of Language*. MIT Press, Cambridge, MA. Reprinted in 1965.

George K. Zipf. 1949. *Human Behavior and the Principle of Least Effort*. Addison-Wesley, Cambridge, MA.

[3] Note that our usage of two-stage AL is slightly different from the original definition in (Becker and Osborne, 2005)

Combining Constituent Parsers

Victoria Fossum
Dept. of Computer Science
University of Michigan
Ann Arbor, MI 48104
vfossum@umich.edu

Kevin Knight
Information Sciences Institute
University of Southern California
Marina del Rey, CA 90292
knight@isi.edu

Abstract

Combining the 1-best output of multiple parsers via parse selection or parse hybridization improves f-score over the best individual parser (Henderson and Brill, 1999; Sagae and Lavie, 2006). We propose three ways to improve upon existing methods for parser combination. First, we propose a method of parse hybridization that recombines *context-free productions* instead of *constituents*, thereby preserving the structure of the output of the individual parsers to a greater extent. Second, we propose an efficient linear-time algorithm for computing expected f-score using Minimum Bayes Risk parse selection. Third, we extend these parser combination methods from multiple 1-best outputs to multiple n-best outputs. We present results on WSJ section 23 and also on the English side of a Chinese-English parallel corpus.

1 Introduction

Parse quality impacts the quality of downstream applications such as syntax-based machine translation (Quirk and Corston-Oliver, 2006). Combining the output of multiple parsers can boost the accuracy of such applications. Parses can be combined in two ways: *parse selection* (selecting the best parse from the output of the individual parsers) or *parse hybridization* (constructing the best parse by recombining sub-sentential components from the output of the individual parsers).

1.1 Related Work

(Henderson and Brill, 1999) perform parse selection by maximizing the expected precision of the selected parse with respect to the set of parses being combined. (Henderson and Brill, 1999) and (Sagae and Lavie, 2006) propose methods for parse hybridization by recombining constituents.

1.2 Our Work

In this work, we propose three ways to improve upon existing methods for parser combination.

First, while constituent recombination (Henderson and Brill, 1999; Sagae and Lavie, 2006) gives a significant improvement in f-score, it tends to flatten the structure of the individual parses. To illustrate, Figures 1 and 2 contrast the output of the Charniak parser with the output of constituent recombination on a sentence from WSJ section 24. We recombine *context-free productions* instead of *constituents*, producing trees containing only context-free productions that have been seen in the individual parsers' output (Figure 3).

Second, the parse selection method of (Henderson and Brill, 1999) selects the parse with maximum expected *precision*; here, we present an efficient, linear-time algorithm for selecting the parse with maximum expected *f-score* within the Minimum Bayes Risk (MBR) framework.

Third, we extend these parser combination methods from 1-best outputs to n-best outputs. We present results on WSJ section 23 and also on the English side of a Chinese-English parallel corpus.

Proceedings of NAACL HLT 2009: Short Papers, pages 253–256,
Boulder, Colorado, June 2009.

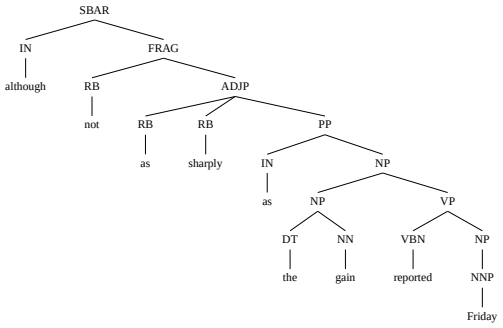


Figure 1: Output of Charniak Parser

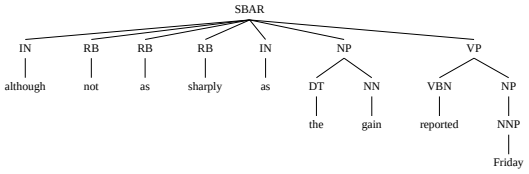


Figure 2: Output of Constituent Recombination

2 Parse Selection

In the MBR framework, although the true reference parse is unknown, we assume that the individual parsers' output forms a reasonable distribution over possible reference parses. We compute the expected f-score of each parse tree p_i using this distribution:

$$expected\ f(p_i) = \sum_{p_j} f(p_i, p_j) \cdot pr(p_j)$$

where $f(p_i, p_j)$ is the f-score of parse p_i with respect to parse p_j and $pr(p_j)$ is the prior probability of parse p_j. We estimate $pr(p_j)$ as follows: $pr(p_j) = pr(parser_k) \cdot pr(p_j|parser_k)$, where $parser_k$ is the parser generating p_j. We set $pr(parser_k)$ according to the proportion of sentences in the development set for which the 1-best output of $parser_k$ achieves the highest f-score of any individual parser, breaking ties randomly.

When $n = 1$, $pr(p_j|parser_k) = 1$ for all p_j; when $n > 1$ we must estimate $pr(p_j|parser_k)$, the distribution over parses in the n-best list output by any given parser. We estimate this distribution using the model score, or log probability, given by $parser_k$ to each entry p_j in its n-best list:

$$pr(p_j|parser_k) = \frac{e^{\alpha * score_{j,k}}}{\sum_{j'=1}^{n} e^{\alpha * score_{j',k}}}$$

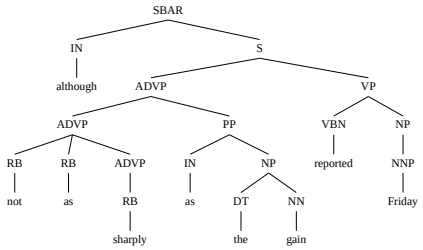


Figure 3: Output of Context-Free Production Recombination

Parser	wsj		ce	
	dev	test	dev	test
Berkeley (Petrov and Klein, 2007)	88.6	89.3	82.9	83.5
Bikel–Collins Model 2 (Bikel, 2002)	87.0	88.2	81.2	80.6
Charniak (Charniak and Johnson, 2005)	90.6	**91.4**	84.7	**84.1**
Soricut–Collins Model 2 (Soricut, 2004)	87.3	88.4	82.3	82.1
Stanford (Klein and Manning, 2003)	85.4	86.4	81.3	80.1

Table 1: F-Scores of 1-best Output of Individual Parsers

We tune α on a development set to maximize f-score,[1] and select the parse p_i with highest expected f-score.

Computing exact expected f-score requires $O(m^2)$ operations per sentence, where m is the number of parses being combined. We can compute an approximate expected f-score in $O(m)$ time. To do so, we compute expected precision for all parses in $O(m)$ time by associating with each unique constituent c_i a list of parses in which it occurs, plus the total probability q_i of those parses. For each parse p associated with c_i, we increment the expected precision of that parse by $q_i/size(p)$. This computation yields the same result as the $O(m^2)$ algorithm. We carry out a similar operation for expected recall. We then compute the harmonic mean of expected precision and expected recall, which closely approximates the true expected f-score.

[1] A low value of α creates a uniform distribution, while a high value concentrates probability mass on the 1-best entry in the n-best list. In practice, tuning α produces a higher f-score than setting α to the value that exactly reproduces the individual parser's probability distribution.

Parse Selection: Minimum Bayes Risk												
System	wsj-dev			wsj-test			ce-dev			ce-test		
	P	R	F	P	R	F	P	R	F	P	R	F
best individual parser	91.3	89.9	90.6	91.8	91.0	91.4	86.1	83.4	84.7	85.6	82.6	84.1
n=1	91.7	90.5	91.1	92.5	91.8	92.0	87.1	84.6	85.8	86.7	83.7	85.2
n=10	92.1	90.8	**91.5**	92.4	91.7	92.0	87.9	85.3	86.6	87.7	84.4	**86.0**
n=25	92.1	90.9	**91.5**	92.4	91.7	92.0	88.0	85.4	**86.7**	87.4	84.2	85.7
n=50	92.1	91.0	**91.5**	92.4	91.7	**92.1**	88.0	85.3	86.6	87.6	84.3	85.9

Table 2: Precision, Recall, and F-score Results from Parse Selection

3 Constituent Recombination

(Henderson and Brill, 1999) convert each parse into constituents with syntactic labels and spans, and weight each constituent by summing $pr(parser_k)$ over all parsers k in whose output the constituent appears. They include all constituents with weight above a threshold $t = \frac{m+1}{2}$, where m is the number of input parses, in the combined parse.

(Sagae and Lavie, 2006) extend this method by tuning t on a development set to maximize f-score.[2] They populate a chart with constituents whose weight meets the threshold, and use a CKY-style parsing algorithm to find the heaviest tree, where the weight of a tree is the sum of its constituents' weights. Parsing is not constrained by a grammar; any context-free production is permitted. Thus, the combined parses may contain context-free productions not seen in the individual parsers' outputs. While this failure to preserve the structure of individual parses does not affect f-score, it may hinder downstream applications.

To extend this method from 1-best to n-best lists, we weight each constituent by summing $pr(parser_k) \cdot pr(p_j|parser_k)$ over all parses p_j generated by $parser_k$ in which the constituent appears.

4 Context-Free Production Recombination

To ensure that all context-free productions in the combined parses have been seen in the individual parsers' outputs, we recombine context-free productions rather than constituents. We convert each parse into context-free productions, labelling each constituent in the production with its span and syntactic category and weighting each production by summing $pr(parser_k) \cdot pr(p_j|parser_k)$ over all parses p_j generated by $parser_k$ in which the production appears. We re-parse the sentence with these productions, returning the heaviest tree (where the weight of a tree is the sum of its context-free productions' weights). We optimize f-score by varying the trade-off between precision and recall using a derivation length penalty, which we tune on a development set.[3]

5 Experiments

Table 1 illustrates the 5 parsers used in our combination experiments and the f-scores of their 1-best output on our data sets. We use the n-best output of the Berkeley, Charniak, and Soricut parsers, and the 1-best output of the Bikel and Stanford parsers. All parsers were trained on the standard WSJ training sections. We use two corpora: the WSJ (sections 24 and 23 are the development and test sets, respectively) and English text from the LDC2007T02 Chinese-English parallel corpus (the development and test sets contain 400 sentences each).

6 Discussion & Conclusion

Results are shown in Tables 2, 3, and 4. On both test sets, constituent recombination achieves the best f-score (1.0 points on WSJ test and 2.3 points on Chinese-English test), followed by context-free production combination, then parse selection, though the differences in f-score among the combination methods are not statistically significant. Increasing the n-best list size from 1 to 10 improves parse selection and context-free production recombination,

[2] A high threshold results in high precision, while a low threshold results in high recall.

[3] By subtracting higher(lower) values of this length penalty from the weight of each production, we can encourage the combination method to favor trees with shorter(longer) derivations and therefore higher precision(recall) at the constituent level.

Parse Hybridization: Constituent Recombination												
System	wsj-dev			wsj-test			ce-dev			ce-test		
	P	R	F	P	R	F	P	R	F	P	R	F
best individual parser	91.3	89.9	90.6	91.8	91.0	91.4	86.1	83.4	84.7	85.6	82.6	84.1
n=1	92.5	90.3	91.4	93.0	91.6	92.3	89.2	84.6	86.8	89.1	83.6	86.2
n=10	92.6	90.5	**91.5**	93.1	91.7	**92.4**	89.9	84.4	**87.1**	89.9	83.2	**86.4**
n=25	92.6	90.5	**91.5**	93.2	91.7	**92.4**	89.9	84.4	87.0	89.7	83.4	**86.4**
n=50	92.6	90.5	**91.5**	93.1	91.7	**92.4**	89.9	84.4	**87.1**	89.7	83.2	86.3

Table 3: Precision, Recall, and F-score Results from Constituent Recombination

Parse Hybridization: Context-Free Production Recombination												
System	wsj-dev			wsj-test			ce-dev			ce-test		
	P	R	F	P	R	F	P	R	F	P	R	F
best individual parser	91.3	89.9	90.6	91.8	91.0	91.4	86.1	83.4	84.7	85.6	82.6	84.1
n=1	91.7	91.0	91.4	92.1	91.9	92.0	86.9	85.4	86.2	86.2	84.3	85.2
n=10	92.1	90.9	91.5	92.5	91.8	**92.2**	87.8	85.1	**86.4**	86.2	84.3	**86.1**
n=25	92.2	91.0	**91.6**	92.5	91.8	**92.2**	87.8	85.1	**86.4**	87.6	84.6	**86.1**
n=50	92.1	90.8	91.4	92.4	91.7	92.1	87.6	84.9	86.2	87.7	84.6	**86.1**

Table 4: Precision, Recall, and F-score Results from Context-Free Production Recombination

though further increasing n does not, in general, help.[4] Chinese-English test set f-score gets a bigger boost from combination than WSJ test set f-score, perhaps because the best individual parser's baseline f-score is lower on the out-of-domain data.

We have presented an algorithm for parse hybridization by recombining context-free productions. While constituent recombination results in the highest f-score of the methods explored, context-free production recombination produces trees which better preserve the syntactic structure of the individual parses. We have also presented an efficient linear-time algorithm for selecting the parse with maximum expected f-score.

Acknowledgments

We thank Steven Abney, John Henderson, and Kenji Sagae for helpful discussions. This research was supported by DARPA (contract HR0011-06-C-0022) and by NSF ITR (grant IIS-0428020).

[4]These diminishing gains in f-score as n increases reflect the diminishing gains in f-score of the oracle parse produced by each individual parser as n increases.

References

Daniel M. Bikel. 2004. *Design of a Multi-lingual, Parallel-processing Statistical Parsing Engine.* In Proceedings of HLT.

Eugene Charniak and Mark Johnson. 2005. *Coarse-to-fine n-best parsing and MaxEnt discriminative reranking*. In Proceedings of ACL.

Michael Collins and Terry Koo. 2005. *Discriminative Reranking for Natural Language Parsing*. Computational Linguistics, 31(1):25-70.

John C. Henderson and Eric Brill. 2000. *Exploiting Diversity in Natural Language Processing: Combining Parsers.* In Proceedings of EMNLP.

Dan Klein and Christopher D. Manning. 2003. *Accurate Unlexicalized Parsing.* In Proceedings of ACL.

Slav Petrov and Dan Klein. 2007. *Improved Inference for Unlexicalized Parsing*. In Proceedings of HLT-NAACL.

Chris Quirk and Simon Corston-Oliver. 2006. *The Impact of Parse Quality on Syntactically-Informed Statistical Machine Translation.* In Proceedings of EMNLP.

Kenji Sagae and Alon Lavie. 2006. *Parser Combination by Reparsing*. In Proceedings of HLT-NAACL.

Radu Soricut. 2004. *A Reimplementation of Collins' Parsing Models*. Technical report, Information Sciences Institute, Department of Computer Science, University of Southern California.

Recognising the Predicate–argument Structure of Tagalog

Meladel Mistica
Australian National University – Linguistics
The University of Melbourne – CSSE
The University of Sydney – Linguistics
mmistica@csse.unimelb.edu.au

Timothy Baldwin
CSSE
The University of Melbourne
tim@csse.unimelb.edu.au

Abstract

This paper describes research on parsing Tagalog text for predicate–argument structure (PAS). We first outline the linguistic phenomenon and corpus annotation process, then detail a series of PAS parsing experiments.

1 Introduction

Predicate–argument structure (PAS) has been shown to be highly valuable in tasks such as information extraction (Surdeanu et al., 2003; Miyao et al., 2009). In this research, we develop a resource for analysing the predicate–argument structure of Tagalog, a free word order language native to the Philippines, and carry out preliminary empirical investigation of PAS parsing methods over Tagalog.

The motivation for this research is the investigation of the interaction between information structure and word order in Tagalog. That is, we wish to determine the utility of discourse-based contextual information in predicting word order in Tagalog, in a natural language generation context. We see PAS as the natural representation for this exploration. This research clearly has implications beyond our immediate interests, however, in terms of resource creation for an NLP resource-poor language, and the facilitation of research on parsing and parsing-based applications in Tagalog. It is also one of the first instances of research on PAS parsing over a genuinely free word order language.

2 Background

Tagalog is an Austronesian language of the Malayo-Polynesian branch, which forms the basis of the national language of the Philippines, Filipino (a.k.a. Pilipino) (Gordon, 2005). It is a verb-initial language, with relatively free word order of verbal arguments (Kroeger, 1993), as exemplified in the word-order variants provided with (1). There are no discernible meaning differences between the provided variants, but there are various soft constraints on free word order, as discussed by Kroeger (1993) and Sells (2000).

(1) *Nagbigay* ***ng*** *libro* ***sa*** *babae* ***ang*** *lalaki*
gave GEN book DAT woman NOM man
"The man gave the woman a book"
Nagbigay ***ng*** *libro* ***ang*** *lalaki* ***sa*** *babae*
Nagbigay ***sa*** *babae* ***ng*** *libro* ***ang*** *lalaki*
Nagbigay ***sa*** *babae* ***ang*** *lalaki* ***ng*** *libro*
Nagbigay ***ang*** *lalaki* ***sa*** *babae* ***ng*** *libro*
Nagbigay ***ang*** *lalaki* ***ng*** *librosa babae*

In addition to these free word order possibilities, Tagalog exhibits *voice marking*, a morpho-syntactic phenomenon which is common in Austronesian languages and gives prominence to an element in a sentence (Schachter and Otanes, 1972; Kroeger, 1993). This poses considerable challenges to generation, because of the combinatorial explosion in the possible ways of expressing what is seemingly the same proposition. Below, we provide a brief introduction to Tagalog syntax, with particular attention to voice marking.

2.1 Constituency

There are three case markers in Tagalog: *ang*, *ng* and *sa*, which are by convention written as separate preposing words, as in (1). These markers normally prepose phrasal arguments of a given verb.

The *sa* marker is predominantly used for goals, recipients, locations and definite objects, while *ng* marks possessors, actors, instruments and indefinite objects (Kroeger, 1993). *Ang* is best explained in terms of Tagalog's *voice-marking* system.

Proceedings of NAACL HLT 2009: Short Papers, pages 257–260,
Boulder, Colorado, June 2009.

2.2 Tagalog Voice Marking

Tagalog has rich verbal morphology which gives prominence to a particular dependent via voice marking (Schachter and Otanes, 1972); this special dependent in the sentence is the *ang*-marked argument.

There are 5 major voice types in Tagalog: Actor Voice (AV); Patient/Object Voice (OV); Dative/Locative Voice (DV); Instrumental Voice (IV); and Benefactive Voice (BV) (Kroeger, 1993). This voice marking, manifested on the verb, reflects the semantic role of the *ang*-marked constituent, as seen in the sentences below from Kroeger (1993), illustrating the 3 voice types of AV, OV, and BV.

(2) **Actor Voice (AV)**

*Bum**ili*** ***ang*** ***lalake*** *ng* *isda* *sa* *tindahan*
buy NOM man GEN fish DAT store
"The man bought fish at the store"

(3) **Object Voice (OV)**

*Bin**ili*** *ng* *lalake* ***ang*** ***isda*** *sa* *tindahan.*
buy GEN man NOM fish DAT store
"The man bought fish at the store"

(4) **Benefactive Voice (BV)**

Ibin**ili *ng* *lalake* *ng* *isda* ***ang*** ***bata.***
buy GEN man GEN fish NOM child
"The man bought fish for the child"

In each case, the morphological marking on the verb (which indicates the voice type) is presented in bold, along with the focused *ang* argument.

In addition to displaying free word order, therefore, Tagalog presents the further choice of which voice to encode the proposition with.

3 Data and Resources

For this research, we annotated our own corpus of Tagalog text for PAS. This is the first such resource to be created for the Tagalog language. To date, we have marked up two chapters (about 2500 tokens) from a narrative obtained from the Gutenberg Project[1] called *Hiwaga ng Pagibig* ("The Mystery of Love"); we intend to expand the amount of annotated data in the future. The annotated data is available from `www.csse.unimelb.edu.au/research/lt/resources/tagalog-pas`.

[1] `http://www.gutenberg.org/etext/18955`

3.1 Part-of-speech Mark-up

First, we developed a set of 5 high-level part-of-speech (POS) tags for the task, with an additional tag for sundries such as punctuation. The tags are as follows:

Description	Example(s)
proper name	names of people/cities
pronoun	personal pronouns
open-class word	nouns, verbs, adjectives
closed-class word	conjunctions
function word	case markers
other	punctuation

These tags are aimed at assisting the identification of constituent boundaries, focusing primarily on differentiating words that have semantic content from those that perform a grammatical function, with the idea that function words, such as case markers, generally mark the start of an argument, while open-class words generally occur within a predicate or argument. Closed-class words, on the other hand (e.g. sentence conjuncts) tend not to be found inside predicates and arguments.

The advantage of having a coarse-grained set of tags is that there is less margin for error and disagreement on how a word can be tagged. For future work, we would like to compare a finer-grained set of tags, such as that employed by dela Vega et al. (2002), with our tags to see if a more detailed distinction results in significant benefits.

In Section 4, we investigate the impact of the inclusion of this extra annotation on PAS recognition, to gauge whether the annotation effort was warranted.

3.2 Predicate and Argument Mark-up

Next, we marked up predicates and their (core) arguments, employing the standard IOB tag scheme. We mark up two types of predicates: PRD and PRD-SUB. The former refers to predicates that belong to main clauses, whilst the latter refers to predicates that occur in subordinate or dependent clauses.

We mark up 4 types of arguments: ANG, NG, SA and NG-COMP. The first three mark nominal

phrases, while the last marks sentential complements (e.g. the object of quotative verbs).

We follow the multi-column format used in the CoNLL 2004 semantic role labelling (SRL) task (Carreras and Màrquez, 2004), with as many columns as there are predicates in a sentence, and one predicate and its associated arguments per column.

3.3 Annotation

Our corpus consists of 259 predicates (47 of which are subordinate, i.e. PRD-SUB), and 435 arguments. The following is a breakdown of the arguments:

Argument type:	SA	ANG	NG	NG-CMP
Count:	83	193	147	12

3.4 Morphological Processing

In tandem with the corpus annotation, we developed a finite-state morphological analyser using XFST and LEXC (Beesley and Karttunen, 2003), that extracts morphological features for individual words in the form of a binary feature vector.[2] While LEXC is ordinarily used to define a lexicon of word stems, we opted instead to list permissible syllables, based on the work of French (1988). This decision was based purely on resource availability: we did not have an extensive list of stems in Tagalog, or the means to generate such a list.

4 Experiments

In this section, we report on preliminary results for PAS recognition over our annotated data. The approach we adopt is similar to the conventional approach adopted in CoNLL-style semantic role labelling: a two-phase approach of first identifying the predicates, then identifying arguments and attaching them to predicates, in a pipeline architecture. Primary areas of investigation in our experiments are: (1) the impact of POS tags on predicate prediction; and (2) the impact of morphological processing on overall performance.

In addition to experimenting with the finite state morphological processing (see Section 3.4), we experiment with a character n-gram method, where we simply take the first and last n characters of a word as features. In our experiments, we set n to 3 and 2 characters for prefix and suffixes, respectively.

We treat each step in the pipeline as a structured learning task, which we model with conditional random fields (Lafferty et al., 2001) using CRF++.[3] All of the results were arrived at via leave-one-out cross-validation, defined at the sentence level, and the evaluation was carried out in terms of precision (P), recall (R) and F-score (F) using the evaluation software from the CoNLL 2004 SRL task.

4.1 Predicate identification

First, we attempt to identify the predicate(s) in a given sentence. Here, we experiment with word context windows of varying width (1–6 words), and also POS features in the given context window. Three different strategies are used to derive the POS tags: (1) from CRF++, with a word bigram context window of width 3 (AUTO1); (2) again from CRF++, with a word bigram context window of width 1 (AUTO2); and (3) from gold-standard POS tags, sourced from the corpus (GOLD). AUTO1 and AUTO2 were the two best-performing POS tagging methods amongst a selection of configurations tested, both achieving a word accuracy of 0.914. We compare these three POS tagging options with a method which uses no POS tag information (NO POS). The results for the different POS taggers with each word context width size are presented in Table 1.

Our results indicate that the optimal window size for the predicate identification is 5 words. We also see that POS contributes to the task, and that the relative difference between the gold-standard POS tags and the best of the automatic POS taggers (AUTO2) is small. Of the two POS taggers, the best performance for AUTO2 is clearly superior to that for AUTO1.

4.2 Argument Identification and Attachment

We next turn to argument identification and attachment, i.e. determining the word extent of arguments which attach to each predicate identified in the first step of the pipeline. Here, we build three predicate recognisers from Section 4.1: NO POS, AUTO2 and

[2] Thanks to Steven Bird for help with infixation and defining permissible syllables for the morphological analyser

[3] http://sourceforge.net/projects/crfpp/

Window size	NO POS			AUTO1			AUTO2			GOLD		
	P	R	F	P	R	F	P	R	F	P	R	F
1	.255	.086	.129	.406	.140	.208	.421	.143	.214	.426	.144	.215
2	.436	.158	.232	.487	.272	.349	.487	.262	.340	.529	.325	.403
3	.500	.190	.275	.477	.255	.332	.500	.262	.344	.571	.335	.422
4	.478	.190	.272	**.509**	**.290**	**.370**	.542	.280	.369	.523	.325	.401
5	**.491**	**.204**	**.278**	.494	.274	.351	**.558**	**.349**	**.429**	**.571**	**.360**	**.442**
6	.478	.190	.272	.484	.269	.346	.490	.262	.341	.547	.338	.418

Table 1: Results for predicate identification (best score in each column in **bold**)

Morphological analysis	NO POS			AUTO2			GOLD		
	P	R	F	P	R	F	P	R	F
FINITE STATE	.362	.137	.199	.407	.201	.269	.420	.207	.278
CHAR n-GRAMS	**.624**	.298	.404	**.643**	.357	**.459**	**.623**	.377	.470
COMBINED	.620	**.307**	**.410**	.599	**.362**	.451	**.623**	**.386**	**.477**

Table 2: Results for argument identification and attachment (best score in each column in **bold**)

GOLD, all based on a window size of 5. We combine these with morphological features from: (1) the finite-state morphological analyser, (2) character n-grams, and (3) the combination of the two. The results of the different combinations are shown in Table 2, all based on a word context window of 3, as this was found to be superior for the task in all cases.

The results with character n-grams were in all cases superior to those for the morphological analyser, although slight gains were seen when the two were combined in most cases (most notably in recall). There was surprisingly little difference between the GOLD results (using gold-standard POS tags) and the AUTO2 results.

5 Conclusion

In this paper, we have presented a system that recognises PAS in Tagalog text. As part of this, we created the first corpus of PAS for Tagalog, and produced preliminary results for predicate identification and argument identification and attachment.

In future work, we would like to experiment with larger datasets, include semantic features, and trial other learners amenable to structured learning tasks.

References

Kenneth R. Beesley and Lauri Karttunen. 2003. *Finite State Morphology*. CSLI Publications, Stanford, USA.

Xavier Carreras and Lluís Màrquez. 2004. Introduction to the CoNLL-2004 shared task: Semantic role labeling. In *Proc. of CoNLL-2004*, pages 89–97, Boston, USA.

Ester D. dela Vega, Melvin Co, and Rowena Cristina Guevara. 2002. Language model for predicting parts of speech of Filipino sentences. In *Proceedings of the 3rd National ECE Conference*.

Koleen Matsuda French. 1988. *Insights into Tagalog*. Summer Institute of Linguistics, Dallas, USA.

Raymond Gordon, Jr. 2005. *Ethnologue: Languages of the World*. SIL International, Dallas, USA, 15th edition.

Paul Kroeger. 1993. *Phrase Structure and Grammatical Relations in Tagalog*. CSLI Publications, Stanford, USA.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML 2001*, pages 282–289, Williamstown, USA.

Yusuke Miyao, Kenji Sagae, Rune Saetre, Takuya Matsuzaki, and Jun'ichi Tsujii. 2009. Evaluating contributions of natural language parsers to protein–protein interaction extraction. *Bioinformatics*, 25(3):394–400.

Paul Schachter and Fe T. Otanes. 1972. *Tagalog Reference Grammar*. University of California Press, Berkeley.

Peter Sells. 2000. Raising and the order of clausal constituents in the Philippine languages. In Ileana Paul, Vivianne Phillips, and Lisa Travis, editors, *Formal Issues in Austronesian Linguistics*, pages 117–143. Kluwer Academic Publishers, Dordrecht, Germany.

Mihai Surdeanu, Sanda Harabagiu, John Williams, and Paul Aarseth. 2003. Using predicate-argument structures for information extraction. In *Proc. of ACL 2003*, pages 8–15, Sapporo, Japan.

Reverse Revision and Linear Tree Combination for Dependency Parsing

Giuseppe Attardi
Dipartimento di Informatica
Università di Pisa
Pisa, Italy
attardi@di.unipi.it

Felice Dell'Orletta
Dipartimento di Informatica
Università di Pisa
Pisa, Italy
felice.dellorletta@di.unipi.it

1 Introduction

Deterministic *transition-based Shift/Reduce* dependency parsers make often mistakes in the analysis of long span dependencies (McDonald & Nivre, 2007).

Titov and Henderson (2007) address this accuracy drop by using a beam search instead of a greedy algorithm for predicting the next parser transition.

We propose a parsing method that allows reducing several of these errors, although maintaining a quasi linear complexity. The method consists in two steps: first the sentence is parsed by a deterministic *Shift/Reduce* parser, then a second deterministic *Shift/Reduce* parser analyzes the sentence in reverse using additional features extracted from the parse trees produced by the first parser.

Right-to-left parsing has been used as part of ensemble-based parsers (Sagae & Lavie, 2006; Hall et al., 2007). Nivre and McDonald (2008) instead use hints from one parse as features in a second parse, exploiting the complementary properties of graph-based parsers (Eisner, 1996; McDonald et al., 2005) and transition-based dependency parsers (Yamada & Matsumoto, 2003; Nivre & Scholz, 2004).

Also our method uses input from a previous parser but only uses parsers of a single type, deterministic transition-based *Shift/Reduce*, maintaining an overall linear complexity. In fact both the ensemble parsers and the stacking solution of Nivre-McDonald involve the computation of the maximum spanning tree (MST) of a graph, which require algorithms of quadratic time complexity (e.g. (Chu & Liu, 1965; Edmonds, 1967)).

We introduce an alternative linear combination method. The algorithm is greedy and works by combining the trees top down. We tested it on the dependency trees produced by three parsers, a *Left-to-Right* (*LR*), a *Right-to-Left* (*RL*) and a *stacked Right-to-Left* parser, or *Reverse Revision* parser (*Rev2*). [1] The experiments show that in practice its output often outperforms the results produced by calculating the MST.

2 Experiments

In the reported experiments we used *DeSR* (Attardi at al., 2007), a freely available implementation of a transition-based parser. The parser processes input tokens advancing on the input with *Shift* actions and accumulates processed tokens on a stack with *Reduce* actions. The parsing algorithm is fully deterministic and linear.

For the *LR* parser and the *Rev2* parser we employed an SVM classifier while a Maximum Entropy classifier, with lower accuracy, was used to create the training set for the *Rev2* parser. The reason for this appears to be that the output of a low accuracy parser with many errors provides a better source of learning to the stacked parser.

The *Rev2* parser exploits the same basic set of features as in the *LR* parser plus the additional features extracted from the output of the *LR* parser listed in Table 1, where: *PHLEMMA* is the lemma of the predicted head, *PHPOS* is the Part of Speech of the predicted head, *PDEP* is the predicted dependency label of a token to its predicted head, *PHDIST* indicates whether a token is located before or after

[1] The *stacked Left-to-Right* parser produced slightly worse results than *Rev2*.

Proceedings of NAACL HLT 2009: Short Papers, pages 261–264,
Boulder, Colorado, June 2009.

Feature	Tokens
PHHLEMMA	$w_0\ w_1$
PHDEP	$w_0\ w_1$
PHPOS	$s_0\ w_0\ w_1$
PHLEMMA	$s_0\ w_0\ w_1$
PDEP	$s_0\ w_0\ w_1$
PHDIST	$s_0\ w_0\ w_1$

Table 1: Additional features used in training the Revision parser.

its predicted head, *PHHLEMMA* is the lemma of the predicted grandparent and *PHDEP* is the predicted dependency label of the predicted head of a token to the predicted grandparent. s_0 refers to a token on top of the stack, w_i refers to word at the i-th relative position with respect to the current word and parsing direction. This feature model was used for all languages in our tests.

We present experiments and comparative error analysis on three representative languages from the CoNLL 2007 shared task (Nivre at al., 2007): Italian, Czech and English. We also report an evaluation on all thirteen languages of the CoNLL-X shared task (Buchholz & Marsi, 2006), for comparison with the results by Nivre and McDonald (2008).

Table 2 shows the Labeled Attachment Score (LAS), for the *Left-to-right* parser (*LR*), *Right-to-Left* (*RL*), *Reverse Revision* parser (*Rev2*), linear parser combination (Comb) and MST parser combination (CombMST).

Figure 1 and 2 present the accuracies of the *LR* and *Rev2* parsers for English relative to the dependency length and the length of sentences, respectively. For Czech and Italian the *RL* parser achieves higher accuracy than the *LR* parser and the *Rev2* parser even higher. The error analysis for Czech showed that the *Rev2* parser improves over the *LR* parser everywhere except in the Recall for dependencies of length between 10 and 14. Such an improvement has positive impact on the analysis of sentences longer than 10 tokens, like for Italian.

2.1 CoNLL-X Results

For direct comparison with the approach by Nivre and McDonald (2008), we present the results on the CoNLL-X corpora (Table 3): MST and MST$_{Malt}$ are the results achieved by the MST parser and the MST parser using hints from Maltparser, Malt and

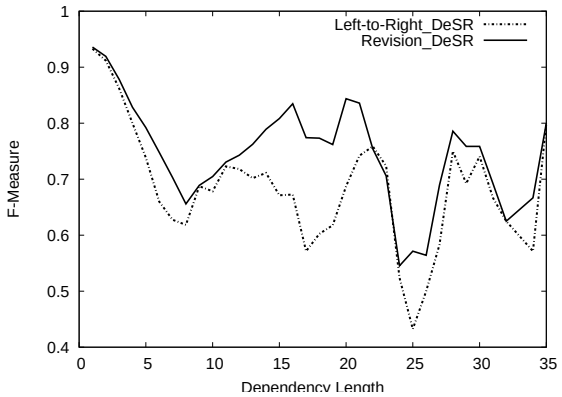


Figure 1: English. F-Measure relative to dependency length.

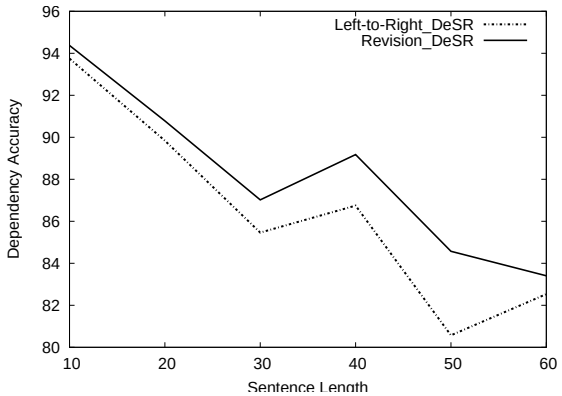


Figure 2: English. Accuracy relative to sentence length.

Malt$_{MST}$ the results of the opposite stacking.

2.2 Remarks

The *Rev2* parser, informed with data from the *LR* parser, achieves better accuracy in twelwe cases, statistically significantly better in eight.

The error analysis confirms that indeed the *Rev2* parser is able to reduce the number of errors made on long dependency links, which are a major weakness of a deterministic Shift/Reduce parser. The accuracy of the *Rev2* parser might be further improved by more sophisticated feature selection, choosing features that better represent hints to the second parsing stage.

3 Linear Voting Combination

Our final improvements arise by combining the outputs of the three parser models: the *LR* parser, the

Language	LR	RL	Rev2	Comb	CombMST	CoNLL 2007 Best
Czech	77.12	78.20	79.95	**80.57**	80.25	80.19
English	86.94	87.44	88.34	89.00	88.79	**89.61**
Italian	81.40	82.89	83.52	**84.56**	84.28	84.40

Table 2: LAS for selected CoNLL 2007 languages.

Language	**LR**	**RL**	**Rev2**	**Comb**	**CombMST**	**Conll-X Best**	**MST**	**MST**$_{Malt}$	**Malt**	**Malt**$_{MST}$
arabic	67.27	66.05	67.54	68.38	68.50	66.91	66.91	**68.64**	66.71	67.80
bulgarian	86.83	87.13	87.41	88.11	87.85	87.57	87.57	**89.05**	87.41	88.59
chinese	87.44	85.77	87.51	87.77	87.75	**89.96**	85.90	88.43	86.92	87.44
czech	79.84	79.46	81.78	82.22	82.22	80.18	80.18	**82.26**	78.42	81.18
danish	83.89	83.63	84.85	85.47	85.25	84.79	84.79	**86.67**	84.77	85.43
dutch	75.71	77.27	78.77	79.55	80.19	79.19	79.19	**81.63**	78.59	79.91
german	85.34	85.20	86.50	87.40	87.38	87.34	87.34	**88.46**	85.82	87.66
japanese	90.03	90.63	90.87	91.67	91.59	91.65	90.71	91.43	91.65	**92.20**
portuguese	86.84	87.00	87.86	88.14	88.20	87.60	86.82	87.50	87.60	**88.64**
slovene	73.64	74.40	75.32	75.72	75.48	73.44	73.44	**75.94**	70.30	74.24
spanish	81.63	81.61	81.85	83.33	83.13	82.25	82.25	**83.99**	81.29	82.41
swedish	82.95	81.62	82.91	83.69	83.69	84.58	82.55	**84.66**	84.58	84.31
turkish	64.91	61.92	63.33	65.27	65.23	65.68	63.19	64.29	65.58	**66.28**
Average	80.49	80.13	81.27	82.05	82.03	81.63	80.83	**82.53**	80.74	82.01

Table 3: Labeled attachment scores for CoNLL-X corpora.

RL parser and the *Rev2* parser.

Instead of using a general algorithm for calculating the MST of a graph, we exploit the fact that we are combining trees and hence we developed an approximate algorithm that has $O(kn)$ complexity, where n is the number of nodes in a tree and k is the number of trees being combined.

The algorithm builds the combined tree T incrementally, starting from the empty tree. We will argue that an invariant of the algorithm is that the partial result T is always a tree.

The algorithm exploits the notion of *fringe* F, i.e. the set of arcs whose parent is in T and that can be added to T without affecting the invariant. Initially F consists of the roots of all trees to be combined. The weight of each arc a in the fringe is the number of parsers that predicted a.

At each step, the algorithm selects from F an arc $a = (h, d, r)$ among those with maximum weight, having $h \in T$. Then it:

1. adds a to T
2. removes from F all arcs whose child is d
3. adds to F all arcs (h', d', r') in the original trees where $h' \in T$ and $d' \notin T$.

Step 3 guarantees that no cycles are present in T. The final T is connected because each added node is connected to a node in T. T is a local maximum because if there were another tree with higher score including arc (h, n, r), either it is present in T or its weight is smaller than the weight for node (h', n, r') in T, as chosen by the algorithm.

The algorithm has $O(kn)$ complexity. A sketch of the proof can be given as follows. Step 3 guarantees that the algorithm is iterated n times, where n is the number of nodes in a component tree. Using appropriate data structures to represent the *fringe* F, insert or delete operations take constant time. At each iteration of the algorithm the maximum number of removals from F (step 2) is constant and it is equal to k, hence the overall cost is $O(nk)$.

Table 2 shows the results for the three languages from CoNLL 2007. With respect to the best results at the CoNLL 2007 Shared Task, the linear parser combination achieves the best LAS for Czech and Italian, the second best for English.

The results for the CoNLL-X languages (Table 3) show also improvements: the *Rev2* parser is more

accurate than MST, except for Bulgarian, Dutch, German, and Spanish, where the difference is within 1%, and it is often better than the Malt$_{MST}$ stacking. The improvements of the *Rev2* over the *LR* parser range from 0.38% for Chinese to 3.84% for Dutch.

The column *CombMST* shows the results of combining parsers using the Chu-Liu-Edmonds MST algorithm and the same weighting scheme of Linear Combination algorithm. For most languages the Linear Combination algorithm leads to a better accuracy than the MST algorithm. The somewhat surprising result might be due indeed to the top down processing of the algorithm: since the algorithm chooses the best among the connections that are higher in the parse tree, this leads to a preference to long spanning links over shorter links even if these contribute higher weights to the MST.

Finally, the run time of the linear combination algorithm on the whole CoNLL-X test set is 11.2 sec, while the MST combination requires 92.5 sec.

We also tested weights based on the accuracy score of each parser for the POS of an arc head, but this produced less accurate results.

4 Conclusions

We presented a method for improving the accuracy of a dependency parser by using a parser that analyzes a sentence in reverse using hints from the trees produced by a forward parser.

We also introduced a new linear algorithm to perform parser combination.

Experiments on the corpora of languages from the CoNLL-X and the CoNLL 2007 shared tasks show that reverse revision parsing improves the accuracy over a transition-based dependency parser in all the tested languages. Further improvements are obtained by using a linear parser combination algorithm on the outputs of three parsers: a *LR* parser, a *RL* parser and a *Rev2* parser.

The combination parser achieves accuracies that are best or second best with respect to the results of the CoNLL 2007 shared task. Since all the individual parsers as well as the combination algorithm is linear, the combined parser maintains an overall linear computational time. On the languages from the CoNLL-X shared task the combination parser achieves often the best accuracy in ten out of thirteen languages but falls short of the accuracy achieved by integrating a graph-based with a transition based parser.

We expect that further tuning of the method might help reduce these differences.

References

G. Attardi, F. Dell'Orletta, M. Simi, A. Chanev and M. Ciaramita. 2007. Multilingual Dependency Parsing and Domain Adaptation using DeSR. In *Proc. of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*.

S. Buchholz and E. Marsi. 2006. CoNLL-X shared task on multilingual dependency parsing. In *Proc. of CoNLL*, 149–164.

Y. J. Chu and T. H. Liu. 1965. On the shortest arborescence of a directed graph. *Science Sinica*(14), 1396–1400.

J. Edmonds. 1967. Optimum branchings. *Journal of Research of the National Bureau of Standards* (71B), 233–240.

J. M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proc. of COLING 1996*, 340–345.

J. Hall, et al. 2007. Single Malt or Blended? A Study in Multilingual Parser Optimization. In *Proc. of the CoNLL Shared Task Session of EMNLP-CoNLL 2007*.

R. McDonald and J. Nivre. 2007. Characterizing the Errors of Data-Driven Dependency Parsing Models In *Proc. of EMNLP-CoNLL 2007*.

R. McDonald, F. Pereira, K. Ribarov and J. Hajič. 2005. Non-projective Dependency Parsing using Spanning Tree Algorithms. In *Proc. of HLT-EMNLP 2005*.

R. McDonald and F. Pereira. 2006. Online Learning of Approximate Dependency Parsing Algorithms. In *Proc. of EACL 2006*.

J. Nivre, et al. 2007. The CoNLL 2007 Shared Task on Dependency Parsing. In *Proc. of the CoNLL Shared Task Session of EMNLP/CoNLL-2007*.

J. Nivre and R. McDonald. 2008. Integrating Graph-Based and Transition-Based Dependency Parsers. In *Proc. of ACL 2008*.

J. Nivre and M. Scholz. 2004. Deterministic Dependency Parsing of English Text. In *Proc. of COLING 2004*.

K. Sagae and A. Lavie. 2006. Parser Combination by Reparsing. In *Proc. of HLT-NAACL 2006*.

I. Titov and J. Henderson. 2007. Fast and Robust Multilingual Dependency Parsing with a Generative Latent Variable Model In *Proc. of the CoNLL Shared Task Session of EMNLP/CoNNL-2007*.

H. Yamada and Y. Matsumoto. 2003. Statistical dependency analysis using support vector machines. In *Proc. of the 8th IWPT*. Nancy, France.

Anchored Speech Recognition for Question Answering

Sibel Yaman[1], **Gokhan Tur**[2], **Dimitra Vergyri**[2], **Dilek Hakkani-Tur**[1],
Mary Harper[3] **and Wen Wang**[2]
[1] International Computer Science Institute
[2] SRI International
[3] Hopkins HLT Center of Excellence, University of Maryland
{sibel,dilek}@icsi.berkeley.edu,{gokhan,dverg,wwang}@speech.sri.com,mharper@casl.umd.edu

Abstract

In this paper, we propose a novel question answering system that searches for responses from spoken documents such as broadcast news stories and conversations. We propose a novel two-step approach, which we refer to as *anchored speech recognition*, to improve the speech recognition of the sentence that supports the answer. In the first step, the sentence that is highly likely to contain the answer is retrieved among the spoken data that has been transcribed using a generic automatic speech recognition (ASR) system. This candidate sentence is then re-recognized in the second step by constraining the ASR search space using the lexical information in the question. Our analysis showed that ASR errors caused a 35% degradation in the performance of the question answering system. Experiments with the proposed anchored recognition approach indicated a significant improvement in the performance of the question answering module, recovering 30% of the answers erroneous due to ASR.

1 Introduction

In this paper, we focus on finding answers to user questions from spoken documents, such as broadcast news stories and conversations. In a typical question answering system, the user query is first processed by an information retrieval (IR) system, which finds out the most relevant documents among massive document collections. Each sentence in these relevant documents is processed to determine whether or not it answers user questions. Once a candidate sentence is determined, it is further processed to extract the exact answer.

Answering factoid questions (i.e., questions like *"What is the capital of France?"*) using web makes use of the redundancy of information (Whittaker et al., 2006). However, when the document collection is not large and when the queries are complex, as in the task we focus on in this paper, more sophisticated syntactic, semantic, and contextual processing of documents and queries is performed to extract or construct the answer. Although much of the work on question answering has been focused on written texts, many emerging systems also enable either spoken queries or spoken document collections (Lamel et al., 2008). The work we describe in this paper also uses spoken data collections to answer user questions but our focus is on improving speech recognition quality of the documents by making use of the wording in the queries. Consider the following example:

Manual transcription: We understand from Greek officials here that it was a Russian-made rocket which is available in many countries but certainly not a weapon used by the Greek military

ASR transcription: to stand firm greek officials here that he was a a russian made rocket uh which is available in many countries but certainly not a weapon used by he great moments

Question: What is certainly not a weapon used by the Greek military?

Answer: a Russian-made rocket

Answering such questions requires as good ASR transcriptions as possible. In many cases, though, there is one generic ASR system and a generic language model to use. The approach proposed in this paper attempts to improve the ASR performance by re-recognizing the candidate sentence using lexical information from the given question. The motiva-

Proceedings of NAACL HLT 2009: Short Papers, pages 265–268,
Boulder, Colorado, June 2009.

tion is that the question and the candidate sentence should share some common words, and therefore the words of the answer sentence can be estimated from the given question. For example, given a factoid question such as: "*What is the tallest building in the world?*", the sentence containing its answer is highly likely to include word sequences such as: "*The tallest building in the world is NAME*" or "*NAME, the highest building in the world, ...*", where NAME is the exact answer.

Once the sentence supporting the answer is located, it is re-recognized such that the candidate answer is constrained to include parts of the question word sequence. To achieve this, a word network is formed to match the answer sentence to the given question. Since the question words are taken as a basis to re-recognize the best-candidate sentence, the question acts as an *anchor*, and therefore, we call this approach *anchored recognition*.

In this work, we restrict out attention to questions about the subject, the object and the locative, temporal, and causative arguments. For instance, the followings are the questions of interest for the sentence *Obama invited Clinton to the White House to discuss the recent developments*:
Who invited Clinton to the White House?
Who did Obama invite to the White House?
Why did Obama invite Clinton to the White House?

2 Sentence Extraction

The goal in sentence extraction is determining the sentence that is most likely to contain the answer to the given question. Our sentence extractor relies on non-stop word n-gram match between the question and the candidate sentence, and returns the sentence with the largest weighted average. Since not all word n-grams have the same importance (e.g. function vs. content words), we perform a weighted sum as typically done in the IR literature, i.e., the matching n-grams are weighted with respect to their inverse document frequency (IDF) and length.

A major concern for accurate sentence extraction is the robustness to speech recognition errors. Another concern is dealing with alternative word sequences for expressing the same meaning. To tackle the second challenge, one can also include synonyms, and compare paraphrases of the question and the candidate answer. Since our main focus is on robustness to speech recognition errors, our data set is limited to those questions that are worded very similarly to the candidate answers. However, the approach is more general, and can be extended to tackle both challenges.

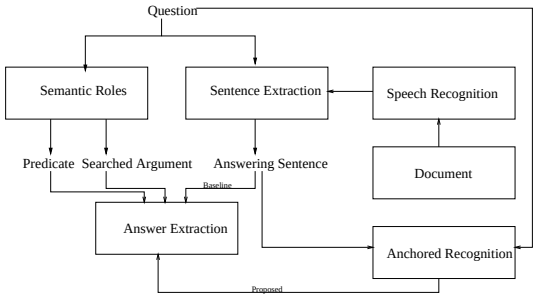


Figure 1: Conceptual scheme of the baseline and proposed information distillation system.

3 Answer Extraction

When the answer is to be extracted from ASR output, the exact answers can be erroneous because (1) the exact answer phrase might be misrecognized, (2) other parts of the sentence might be misrecognized, so the exact answer cannot be extracted either because parser fails or because the sentence cannot match the query.

The question in the example in the *Introduction* section is concerned with the object of the predicate "*is*" rather than of the other predicates "*understand*" or "*was*". Therefore, a pre-processing step is needed to correctly identify the object (in this example) that is being asked, which is described next.

Once the best candidate sentence is estimated, a syntactic parser (Harper and Huang,) that also outputs *function tags* is used to parse both the question and candidate answering sentence. The parser is trained on Fisher, Switchboard, and speechified Broadcast News, Brown, and Wall Street Journal treebanks without punctuation and case to match input the evaluation conditions.

An example of such a syntactic parse is given in Figure 2. As shown there, the "SBJ" marks the surface subject of a given predicate, and the "TMP" tag marks the temporal argument. There are also the "DIR" and "LOC" tags indicating the locative argument and the "PRP" tag indicating the causal argument. Such parses not only provide a mechanism to extract information relating to the subject of the predicate of interest, but also to extract the part of

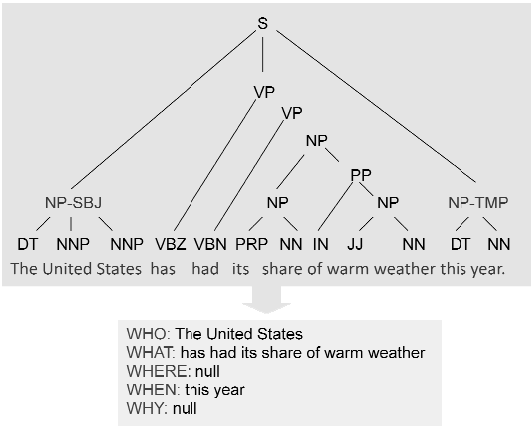


Figure 2: The function tags assist in finding the subject, object, and arguments of a given predicate.

the sentence that the question is about, in this example "*a Russian-made rocket [which] is certainly not a weapon used by the Greek military*". The extraction of the relevant part is achieved by matching the predicate of the question to the predicates of the subsentences in the best candidate sentence. Once such syntactic parses are obtained for the part of the best-candidate sentence that matches the question, a set of rules are used to extract the argument that can answer the question.

4 Anchored Speech Recognition

In this study we employed a state-of-the-art broadcast news and conversations speech recognition system (Stolcke et al., 2006). The recognizer performs a total of seven decoding passes with alternating acoustic front-ends: one based on Mel frequency cepstral coefficients (MFCCs) augmented with discriminatively estimated multilayer-perceptron (MLP) features, and one based on perceptual linear prediction (PLP) features. Acoustic models are cross-adapted during recognition to output from previous recognition stages, and the output of the three final decoding steps are combined via confusion networks.

Given a question whose answer we expect to find in a given sentence, we construct a re-decoding network to match that question. We call this process *anchored speech recognition*, where the anchor is the question text. Note that this is different than forced alignment, which enforces the recognition of an audio stream to align with some given sentence. It is used for detecting the start times of individual words or for language learning applications to exploit the acoustic model scores, since there is no need for a language model.

Our approach is also different than the so-called *flexible alignment* (Finke and Waibel, 1997), which is basically forced alignment that allows skipping any part of the given sentence, replacing it with a reject token, or inserting hesitations in between words. In our task, we require all the words in the question to be in the best-candidate sentence without any skips or insertions. If we allow flexible alignment, then any part of the question could be deleted. In the proposed anchored speech recognition scheme, we allow only pauses and rejects between words, but do not allow any deletions or skips.

The algorithm for extracting anchored recognition hypotheses is as follows: (i) Construct new recognition and rescoring language models (LMs) by interpolating the baseline LMs with those trained from only the question sentences and use the new LM to generate lattices - this aims to bias the recognition towards word phrases that are included in the questions. (ii) Construct for each question an "anchored" word network that matches the word sequence of the question, allowing any other word sequence around it. For example if the question is *WHAT did Bruce Gordon say?*, we construct a word network to match *Bruce Gordon said ANYTHING* where "ANYTHING" is a filler that allows any word (a word loop). (iii) Intersect the recognition lattices from step (i) with the anchored network for each question in (ii), thus extracting from the lattice only the paths that match as answers to the question. Then rescore that new lattice with higher order LM and cross-word adapted acoustic models to get the best path. (iv) If the intersection part in (iii) fails then we use a more constrained recognition network: Starting with the anchored network in (ii) we first limit the vocabulary in the ANYTHING word-loop sub-network to only the words that were included in the recognition lattice from step (i). Then we compose this network with the bigram LM (from step (i)) to add bigram probabilities to the network. Vocabulary limitation is done for efficiency reasons. We also allow optional filler words and pauses to this network to allow for hesitations, non-speech events and pauses within the utterance we are trying to match. This may limit somewhat the potential improvement from this approach and we are working

Question Type	ASR Output	Manual Trans.
Subject	85%	98%
Object	75%	90%
Locative Arg.	81%	93%
Temporal Arg.	94%	98%
Reason	86%	100%
Total	83%	95%

Table 1: Performance figures for the sentence extraction system using automatic and manual transcriptions.

Question Type	ASR Output	Manual Trans.	Anchored Output
Subject	51%	77%	61%
Object	41%	73%	51%
Locative Arg.	18%	22%	22%
Temporal Arg.	55%	73%	63%
Reason	26%	47%	26%
Total	44%	68%	52%

Table 2: Performance figures for the answer extraction system using automatic and manual transcriptions compared with anchored recognition outputs.

towards enhancing the vocabulary with more candidate words that could contain the spoken words in the region. (v) Then we perform recognition with the new anchored network and extract the best path through it. Thus we enforce partial alignment of the audio with the question given, while the regular recognition LM is still used for the parts outside the question.

5 Experiments and Results

We performed experiments using a set of questions and broadcast audio documents released by LDC for the DARPA-funded GALE project Phase 3. In this dataset we have 482 questions (177 subject, 160 object, 73 temporal argument, 49 locative argument, and 23 reason) from 90 documents. The ASR word error rate (WER) for the sentences from which the questions are constructed is 37% with respect to noisy closed captions. To factor out IR noise we assumed that the target document is given.

Table 1 presents the performance of the sentence extraction system using manual and automatic transcriptions. As seen, the system is almost perfect when there is no noise, however performance degrades about 12% with the ASR output.

The next set of experiments demonstrate the performance of the answer extraction system when the correct sentence is given using both automatic and manual transcriptions. As seen from Table 2, the answer extraction performance degrades by about 35% relative using the ASR output. However, using the anchored recognition approach, this improves to 23%, reducing the effect of the ASR noise significantly[1] by more than 30% relative. This is shown in the last column of this table, demonstrating the use of the proposed approach. We observe that the WER of the sentences for which we now get corrected answers is reduced from 45% to 28% with this approach, a reduction of 37% relative.

6 Conclusions

We have presented a question answering system for querying spoken documents with a novel anchored speech recognition approach, which aims to re-decode an utterance given the question. The proposed approach significantly lowers the error rate for answer extraction. Our future work involves handling audio in foreign languages, that is robust to both ASR and machine translation noise.

Acknowledgments: This work was funded by DARPA under contract No. HR0011-06-C-0023. Any conclusions or recommendations are those of the authors and do not necessarily reflect the views of DARPA.

References

M. Finke and A. Waibel. 1997. Flexible transcription alignment. In *Proceedings of the IEEE ASRU Workshop*, Santa Barbara, CA.

M. Harper and Z. Huang. *Chinese Statistical Parsing*, chapter To appear.

L. Lamel, S. Rosset, C. Ayache, D. Mostefa, J. Turmo, and P. Comas. 2008. Question answering on speech transcriptions: the qast evaluation in clef. In *Proceedings of the LREC*, Marrakech, Morocco.

A. Stolcke, B. Chen, H. Franco, V. R. R. Gadde, M. Graciarena, M.-Y. Hwang, K. Kirchhoff, N. Morgan, X. Lin, 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. *IEEE Transactions on Audio, Speech, and Language Processing*, 14(5):1729–1744, September.

E. W. D. Whittaker, J. Mrozinski, and S. Furui. 2006. Factoid question answering with web, mobile and speech interfaces. In *Proceedings of the NAACL/HLT*, Morristown, NJ.

[1] according to the Z-test with 0.95 confidence interval

Score Distribution Based Term Specific Thresholding for Spoken Term Detection

Doğan Can and **Murat Saraçlar**
Electrical & Electronics Engineering Department
Boğaziçi University
İstanbul, Turkey
{dogan.can, murat.saraclar}@boun.edu.tr

Abstract

The spoken term detection (STD) task aims to return relevant segments from a spoken archive that contain the query terms. This paper focuses on the decision stage of an STD system. We propose a term specific thresholding (TST) method that uses per query posterior score distributions. The STD system described in this paper indexes word-level lattices produced by an LVCSR system using Weighted Finite State Transducers (WFSTs). The target application is a sign dictionary where precision is more important than recall. Experiments compare the performance of different thresholding techniques. The proposed approach increases the maximum precision attainable by the system.

1 Introduction

The availability of vast multimedia archives calls for solutions to efficiently search this data. Multimedia content also enables interesting applications which utilize multiple modalities, such as speech and video. Spoken term detection (STD) is a subfield of speech retrieval, which locates occurrences of a query in a spoken archive. In this work, STD is used as a tool to segment and retrieve the signs in news videos for the hearing impaired based on speech information. After the location of the query is extracted with STD, the sign video corresponding to that time interval is displayed to the user. In addition to being used as a sign language dictionary this approach can also be used to automatically create annotated sign databases that can be utilized for training sign recognizers (Aran et al., 2008). For these applications the precision of the system is more important than its recall.

The classical STD approach consists of converting the speech to word transcripts using large vocabulary continuous speech recognition (LVCSR) tools and extending classical information retrieval techniques to word transcripts. However, retrieval performance is highly dependent on the recognition errors. In this context, lattice indexing provides a means of reducing the effect of recognition errors by incorporating alternative transcriptions in a probabilistic framework. A system using lattices can also return the posterior probability of a query as a detection score. Various operating points can be obtained by comparing the detection scores to a threshold. In addition to using a global detection threshold, choosing term specific thresholds that optimize the STD evaluation metric known as Term-Weighted Value (TWV) was recently proposed (Miller et al., 2007). A similar approach which trains a neural network mapping various features to the target classes was used in (Vergyri et al., 2007).

The rest of the paper is organized as follows. In Section 2 we explain the methods used for spoken term detection. These include the indexing and search framework based on WFSTs and the detection framework based on posterior score distributions. In Section 3 we describe our experimental setup and present the results. Finally, in Section 4 we summarize our contributions and discuss possible future directions.

Proceedings of NAACL HLT 2009: Short Papers, pages 269–272,
Boulder, Colorado, June 2009.

2 Methods

The STD system used in this study consists of four stages. In the first stage, an LVCSR system is used to generate lattices from speech. In the second stage the lattices are indexed for efficient retrieval. When a query is presented to the system a set of candidates ranked by posterior probabilities are obtained from the index. In the final stage, the posterior probabilities are compared to a threshold to decide which candidates should be returned.

2.1 Indexing and Retrieval using Finite-State Automata

General indexation of weighted automata (Allauzen et al., 2004) provides an efficient means of indexing for STD (Parlak and Saraçlar, 2008; Can et al., 2009), where retrieval is based on the posterior probability of a term in a given time interval. In this work, the weighted automata to be indexed are the preprocessed lattice outputs of the ASR system. The input labels are phones, the output labels are quantized time-intervals and the weights are normalized negative log probabilities. The index is represented as a WFST where each substring (factor) leads to a successful path over the input labels whenever that particular substring was observed. Output labels of these paths carry the time interval information followed by the utterance IDs. The path weights give the probability of each factor occurring in the specific time interval of that utterance. The index is optimized by WFST determinization and minimization so that the search complexity is linear in the sum of the query length and the number of times the query appears in the index.

2.2 Decision Mechanism

Once a list of candidates ranked with respect to their posterior probabilities are determined using the index, the candidates exceeding a threshold are returned by the system. The threshold is computed to minimize the Bayes risk. In this framework, we need to specify a cost function, prior probabilities and likelihood functions for each class. We choose the cost of a miss to be 1 and the cost of a false alarm to be a free parameter, α. The prior probabilities and the likelihood functions are estimated from the posterior scores of the candidate results for each query.

The likelihood functions are found by fitting parametric models to the score distributions (Manmatha et al., 2001). In this study, the score distributions are modeled by exponential distributions. When the system returns a score, we do not know whether it belongs to the correct or incorrect group, so we use a mixture of two exponential distributions to model the posterior scores returned by the system. The exponential mixture model (EMM) parameters are determined via unsupervised estimation using the Expectation-Maximization (EM) algorithm. Figure 1 shows the normalized histogram of posterior scores and the EM estimate given by our method for an example query.

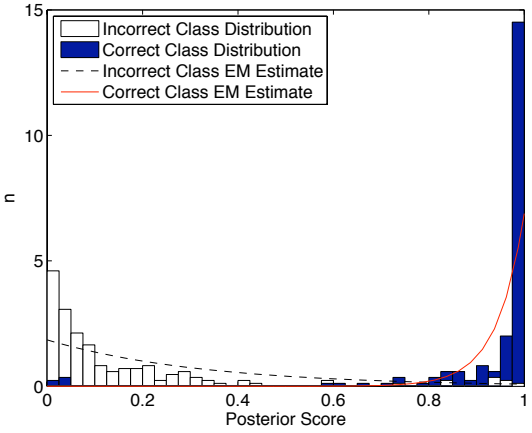


Figure 1: The normalized histogram of posterior scores and the EM estimates for correct and incorrect detections given an example query.

If we denote the posterior score of each candidate by x, incorrect class by c_0 and correct class by c_1, we have

$$p(x) = P(c_0)p(x|c_0) + P(c_1)p(x|c_1)$$

where the incorrect class likelihood $p(x|c_0) = \lambda_0 e^{-\lambda_0 x}$ and correct class likelihood $p(x|c_1) = \lambda_1 e^{-\lambda_1(1-x)}$. The model parameters $\lambda_0, \lambda_1, P(c_0), P(c_1)$ are estimated using the EM algorithm given the scores x_i for $i = 1, \ldots, N$. Each iteration consists of first computing $P(c_j|x_i) = P(c_j)p(x_i|c_j)/p(x_i)$ for $j = 1, 2$ and then updating

$$P(c_j) = \frac{1}{N} \sum_i P(c_j|x_i),$$

$$\lambda_0 = \frac{\sum_i P(c_0|x_i)}{\sum_i P(c_0|x_i)x_i},$$

$$\lambda_1 = \frac{\sum_i P(c_1|x_i)}{\sum_i P(c_1|x_i)(1-x_i)}.$$

After the mixture parameters are estimated, we assume that each mixture represents a class and mixture weights correspond to class priors. Then, the Minimum Bayes Risk (MBR) detection threshold for x is given as:

$$\frac{\lambda_1 + \log(\lambda_0/\lambda_1) + \log(P(c_0)/P(c_1)) + \log\alpha}{\lambda_0 + \lambda_1}.$$

3 Experiments

3.1 Data and Application

Turkish Radio and Television Channel 2 (TRT2) broadcasts a news program for the hearing impaired which contains speech as well as signs. We have collected 11 hours (total speech time) of test material from this broadcast and performed our experiments on this data with a total of 10229 single word queries extracted from the reference transcriptions. We used IBM Attila speech recognition toolkit (Soltau et al., 2007) at the back-end of our system to produce recognition lattices. The ASR system is trained on 100 hours of speech and transcription data collected from various TV and radio broadcasts including TRT2 hearing impaired news, and a general text corpus of size 100 million words.

Our application uses the speech modality to retrieve the signs corresponding to a text query. Retrieved results are displayed as video demonstrations to support the learning of sign language. Since the application acts like an interactive dictionary of sign language, primary concern is to return correct results no matter how few they are. Thus high precision is appreciated much more than high recall rates.

3.2 Evaluation Measures

In our experiments, we use precision and recall as the primary evaluation metrics. For a set of queries $q_k, k = 1,\ldots,Q$,

$$\text{Precision} = \frac{1}{Q}\sum_k \frac{C(q_k)}{A(q_k)} \quad \text{Recall} = \frac{1}{Q}\sum_k \frac{C(q_k)}{R(q_k)}$$

where:
$R(q_k)$: Number of occurences of query q_k,
$A(q_k)$: Total no. of retrieved documents for q_k,
$C(q_k)$: No. of correctly retrieved documents for q_k.

We obtain a precision/recall curve by changing the free parameter associated with each thresholding method to simulate different decision cost settings. Right end of these curves fall into the high precision region which is the main concern in our application.

For the case of global thresholding (GT), the same threshold θ is used for all queries. TWV based term specific thresholding (TWV-TST) (Miller et al., 2007) aims to maximize the TWV metric introduced during NIST 2006 STD Evaluations (NIST, 2006).

$$\text{TWV} = 1 - \frac{1}{Q}\sum_{k=1}^{Q}\{P_{miss}(q_k) + \beta.P_{FA}(q_k)\}$$

$$\text{P}_{\text{miss}}(q_k) = 1 - \frac{C(q_k)}{R(q_k)}, \text{P}_{\text{FA}}(\text{q}_\text{k}) = \frac{A(q_k) - C(q_k)}{T - C(q_k)}$$

where T is the total duration of the speech archive and β is a weight assigned to false alarms that is proportional to the prior probability of occurence of a specific term and its cost-value ratio. This method sets individual thresholds for each query term considering per query expected counts and the tuning parameter β. In the proposed method α plays the same role as β and allows us to control the decision threshold for different cost settings.

3.3 Results

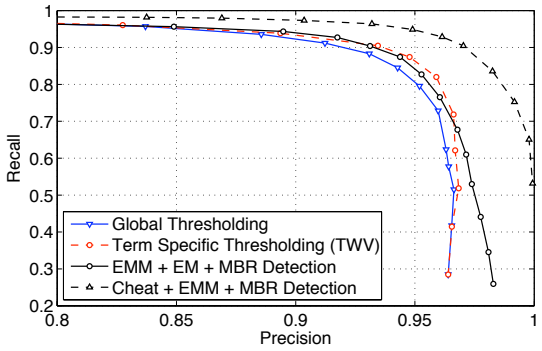


Figure 2: The precision and recall curves for various thresholding techniques.

Figure 2 compares GT, TWV-TST, and the proposed method that utilizes score distributions to derive an optimal decision threshold. For GT and TWT-TST, last precision/recall point in the figure corresponds to the limit threshold value which is 1.0. Both the TWV-TST and the proposed method outperform GT over the entire region of interest. While TWV-TST provides better performance around the

knees of the curves, proposed method achieves higher maximum precision values which coincides with the primary objective of our application.

Figure 2 also provides a curve of what happens when the correct class labels are used to estimate the parameters of the exponential mixture model in a supervised manner instead of using EM. This curve provides an upper bound on the performance of the proposed method.

4 Discussion

In this paper, we proposed a TST scheme for STD which works almost as good as TWV-TST. Extrapolating from the cheating experiment, we believe that the proposed method has potential for outperforming the TWV-TST over the entire region of interest given better initial estimates for the correct and incorrect classes.

A special remark goes to the performance in the high precision region where our method clearly outperforms the rest. While GT and TWV-TST methods are bounded around 96.5% precision value, our method reaches at higher precision figures. For GT, this behavior is due to the inability to set different thresholds for different queries. For TWT-TST, in the high precision region where β is large, the threshold is very close to 1.0 value no matter what the expected count of the query term is, thus it essentially acts like a global threshold.

Our current implementation of the proposed method does not make use of training data to estimate the initial parameters for the EM algorithm. Instead, it relies on some loose assumptions about the initial parameters of the likelihood functions and uses uninformative prior distributions. The significant difference between the upper bound and the actual performance of the proposed method indicates that the current implementation can be improved by better initial estimates.

Our assumption about the parametric form of the likelihood function may not be valid at all times. Maximizing the likelihood with mismatched models degrades the performance even when initial parameters are close to the optimal values. In the future, other parametric forms can be utilized to better model the posterior score distributions.

Maximum likelihood estimation with insufficient data is prone to overtraining. This is a common situation with the STD task at hand. With the current data, three or less results are returned for half of the queries. Bayesian methods can be used to introduce priors on the model parameters in order to make the estimation more robust.

Acknowledgments

This study was supported in part by Boğaziçi University Research Fund (BAP) under the project number 05HA202, TÜBİTAK under the project number 105E102 and Turkish State Planning Organization (DPT) under the project number DPT2007K120610.

References

C. Allauzen, M. Mohri, and M. Saraçlar. 2004. General-indexation of weighted automata-application to spoken utterance retrieval. In *Proc. Workshop on Interdisciplinary Approaches to Speech Indexing and Retrieval at HLT-NAACL*, pages 33–40, March.

O. Aran, I. Arı, E. Dikici, S. Parlak, P. Campr, M. Hruz, L. Akarun, and M. Saraçlar. 2008. Speech and sliding text aided sign retrieval from hearing impaired sign news videos. *Journal on Multimodal User Interfaces*, 2(1):117–131, November.

D. Can, E. Cooper, A. Sethy, C.M. White, B. Ramabhadran, and M. Saraclar. 2009. Effect of pronunciations on oov queries in spoken term detection. In *ICASSP*, April.

R. Manmatha, T. Rath, and F. Feng. 2001. Modeling score distributions for combining the outputs of search engines. In *SIGIR '01*, pages 267–275, New York, NY, USA. ACM.

D. R. H. Miller, M. Kleber, C. Kao, O. Kimball, T. Colthurst, S. A. Lowe, R. M. Schwartz, and H. Gish. 2007. Rapid and accurate spoken term detection. In *Proc. Interspeech*, pages 314–317, August.

NIST. 2006. The spoken term detection (STD) 2006 evaluation plan http://www.nist.gov/speech/tests/std/.

S. Parlak and M. Saraçlar. 2008. Spoken term detection for Turkish broadcast news. In *Proc. ICASSP*, pages 5244–5247, April.

H. Soltau, G. Saon, D. Povey, L. Mangu, J. Kuo, M. Omar, and G. Zweig. 2007. The IBM 2006 GALE Arabic ASR system. In *Proc. ICASSP 2007*, Honolulu, HI, USA.

D. Vergyri, I. Shafran, A. Stolcke, R. R. Gadde, M. Akbacak, B. Roark, and W. Wang. 2007. The SRI/OGI 2006 spoken term detection system. In *Proc. Interspeech*, pages 2393–2396, August.

Automatic Chinese Abbreviation Generation Using Conditional Random Field

Dong Yang, Yi-cheng Pan, and Sadaoki Furui
Department of Computer Science
Tokyo Institute of Technology
Tokyo 152-8552 Japan
{raymond,thomas,furui}@furui.cs.titech.ac.jp

Abstract

This paper presents a new method for automatically generating abbreviations for Chinese organization names. Abbreviations are commonly used in spoken Chinese, especially for organization names. The generation of Chinese abbreviation is much more complex than English abbreviations, most of which are acronyms and truncations. The abbreviation generation process is formulated as a character tagging problem and the conditional random field (CRF) is used as the tagging model. A carefully selected group of features is used in the CRF model. After generating a list of abbreviation candidates using the CRF, a length model is incorporated to re-rank the candidates. Finally the full-name and abbreviation co-occurrence information from a web search engine is utilized to further improve the performance. We achieved top-10 coverage of 88.3% by the proposed method.

1 Introduction

Long named entities are frequently abbreviated in oral Chinese language for efficiency and simplicity. Therefore, abbreviation modeling is an important building component for many systems that accept spoken input, such as directory assistance and voice search systems.

While English abbreviations are usually formed as acronyms, Chinese abbreviations are much more complex, as shown in Figure 1. Most of the Chinese abbreviations are formed by selecting several characters from full-names, which are not necessarily the first character of each word. Usually the original character order in the full-name is preserved in the abbreviation. However, re-ordering of characters as shown in the third example in Figure 1 where characters "三" and "医" are swapped in the abbreviation, also happens.

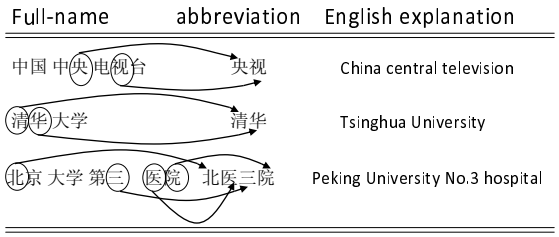


Figure 1: Chinese abbreviation examples

There has been a considerable amount of research on extracting full-name and abbreviation pairs in the same document for obtaining abbreviations (Li and Yarowsky, 2008; Sun et al., 2006; Fu et al., 2006). However, generation of abbreviations given a full-name is still a non-trivial problem. Chang and Lai (Chang and Lai, 2004) have proposed using a hidden Markov model to generate abbreviations from full-names. However, their method assumes that there is no word-to-null mapping, which means that every word in the full-name has to contribute at least one character to the abbreviation. This assumption does not hold for organizations' names which have many word skips in the abbreviation generation.

The CRF was first introduced to natural language processing (NLP) by (Lafferty et al., 2001) and has been widely used in word segmentation, part-of-speech (POS) tagging, and some other NLP tasks. In this paper, we convert the Chinese abbreviation generation process to a CRF tagging problem. The key problem here is how to find a group of discrim-

Proceedings of NAACL HLT 2009: Short Papers, pages 273–276,
Boulder, Colorado, June 2009.

inant and robust features. After using the CRF, we get a list of abbreviation candidates with associate probability scores. We also use the prior conditional probability of the length of the abbreviations given the length of the full-names to complement the CRF probability scores. Such global information is hard to include in the CRF model. In addition, we apply the full-name and abbreviation candidate co-occurrence statistics obtained on the web to increase the correctness of the abbreviation candidates.

2 Chinese Abbreviation Introduction

Chinese abbreviations are generated by three methods (Lee, 2005): reduction, elimination, and generalization.

Both in the reduction and elimination methods, characters are selected from the full-name, and the order of the characters is sometime changed. Note that this paper does not cover the case when the order is changed. The elimination means that one or more words in the full-name are ignored completely, while the reduction requires that at least one character is selected from each word. All the three examples in Figure 1 are produced by the elimination, where at least one word is skipped.

Generalization, which is used to abbreviate a list of similar terms, is usually composed of the number of terms and a shared character across the terms. A example is "三军" (three forces) for "陆军，海军，空军" (land force, sea force, air force). This is the most difficult scenario for the abbreviations and is not considered in this paper.

3 CRF Model for Abbreviation Modeling

3.1 CRF model

A CRF is an undirected graphical model and assigns the following probability to a label sequence $L = l_1 l_2 \dots l_T$, given an input sequence $C = c_1 c_2 \dots c_T$,

$$P(L|C) = \frac{1}{Z(C)} exp(\sum_{t=1}^{T} \sum_{k} \lambda_k f_k(l_t, l_{t-1}, C, t)) \quad (1)$$

Here, f_k is the feature function for the *k*-th feature, λ_k is the parameter which controls the weight of the *k*-th feature in the model, and *Z*(*C*) is the normalization term that makes the summation of the probability of all label sequences to 1. CRF training is usually performed through the typical L-BFGS algorithm (Wallach, 2002) and decoding is performed by Viterbi algorithm (Viterbi, 1967). In this paper, we use an open source toolkit "crf++".

3.2 Abbreviation modeling as a tagging problem

In order to use the CRF method in abbreviation generation, the abbreviation generation problem was converted to a tagging problem. The character is used as a tagging unit and each character in a full-name is tagged by a binary variable with the values of either Y or N: Y stands for a character used in the abbreviation and N means not. An example is given in Figure 2.

Figure 2: Abbreviation in the CRF tagging format

3.3 Feature selection for the CRF

In the CRF method, feature function describes a co-occurrence relation, and it is defined as $f_k(l_t, l_{t-1}, C, t)$ (Eq. 1). f_k is usually a binary function, and takes the value 1 when both observation c_t and transition $l_{t-1} \rightarrow l_t$ are observed. In our abbreviation generation model, we use the following features:

1. Current character The character itself is the most important feature for abbreviation as it will be either retained or discarded. For example, "局" (bureau) and "所" (institue), indicating a government department, are very common characters used in abbreviations. When they appear in full-names, they are likely to be kept in abbreviations.

2. Current word In the full name of "中国农业大学" (China Agricultural university), the word "中国" (China) is usually ignored in the abbreviation, but the word "农业" (agriculture) is usually kept. The length (the number of characters) is also an important feature of the current word.

3. Position of the current character in the current word Previous work (Chang and Lai, 2004) showed that the first character of a word has high possibility to form part of the abbreviation and this is also true for the last character of a three-character word.

4. Combination of feature 2. and 3. above Combination of the features 2 and 3 is expected to improve the performance, since the position infor-

mation affects the abbreviation along with the current word. For example, ending character in "大学" (university) and that in "研究院" (research institute) have very different possibilities to be selected for abbreviations.

Besides the features above, we have examined context information (previous word, previous character, next character, etc.) and other local features like the length of the word, but these features did not improve the performance. The reason may be due to the sparseness of the training data.

4 Improvement by a Length Model and a Web Search Engine

4.1 Length model

There is a strong correlation between the length of organizations' full-names and their abbreviations. We use the length modeling based on discrete probability of $P(M|L)$, in which the variables *M* and *L* are lengths of abbreviations and full-names, respectively. Since it is difficult to incorporate length information into the CRF model explicitly, we use $P(M|L)$ to rescore the output of the CRF.

In order to use the length information, we model the abbreviation process with two steps:

- 1st step: evaluate the length in abbreviation according to the length model $P(M|L)$;
- 2nd step: choose the abbreviation, given the length and full-name.

We assume the following approximation:

$$P(A|F) \simeq P(M|L) \cdot P(A|M,F) \qquad (2)$$

in which variable *A* is the abbreviation and *F* is the full-name; $P(M|L)$ is the length model, and the second probability can be calculated according to the Bayesian rule:

$$P(A|M,F) = \frac{P(A,M|F)}{P(M|F)} = \frac{P(A,M|F)}{\sum_{length(A')=M} P(A',M|F)} \qquad (3)$$

It is obvious that $P(A,M|F) = P(A|F)$ (as *A* contains the information *M* implicitly) and $P(A|F)$ can be obtained from the output of the CRF.

4.2 Web search engine

Co-occurrence of a full-name and an abbreviation candidate can be a clue of the correctness of the abbreviation. We use the "abbreviation candidate"+ "full-name" as queries and input them to the most popular Chinese search engine (www.baidu.com), and then we use the number of hits as the metric to perform re-ranking. The hits is theoretically related to the number of pages which contain both the full-name and abbreviation. The bigger the value of hits, the higher probability that the abbreviation is correct.

We then simply multiply the previous probability score, obtained from Eq. 2, by the number of hits and re-rank the top-30 candidates accordingly.

There are some other ways to use information retrieval methods (Mandala et al., 2000). Our method has an advantage that the access load to the web search engine is relatively small.

5 Experiment

5.1 Data introduction

The corpus we use in this paper comes from two sources: one is the book "modern Chinese abbreviation dictionary" (Yuan and Ruan, 2002) and the other is the wikipedia. Altogether we collected 1945 pairs of organization full-names and their abbreviations.

The data is randomly divided into two parts, a training set with 1298 pairs and a test set with 647 pairs. Table 1 shows the length mapping statistics of the training set. It can be seen that the average length of full-names is about 7.29. We know that for a full-name with length *N*, the number of abbreviation candidates is about $2^N - 2 - N$ (exclude length of 0, 1, and *N*) and we can conclude that the average number of candidates for organization names in this corpus is more than 100.

5.2 Results

The abbreviation method described is part of a project to develop a voice-based search application. For our name abbreviation system we plan to add 10 abbreviation candidates for each organization name into the vocabulary of our voice search application, hence here we consider top-10 coverage.

length of full-name	length of abbreviation					sum
	2	3	4	5	>5	
4	107	1	0	0	0	108
5	89	140	0	0	0	229
6	96	45	46	0	0	187
7	60	189	49	16	0	314
8	48	29	60	3	6	146
9	10	47	35	12	2	106
10	18	11	29	8	6	73
others	21	43	38	17	14	133
average length of the full-name						**7.27**
average length of the abbreviation						**3.01**

Table 1: Length statistics on the training set

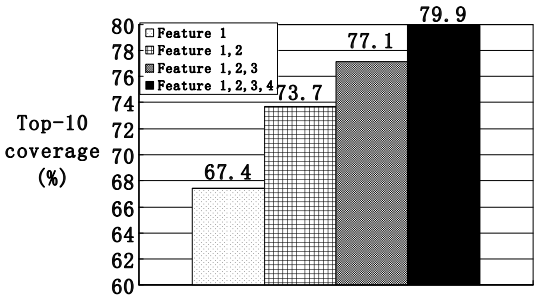


Figure 3: Contribution of features in CRF

Figure 3 shows the result for various combinations of features introduced in Section 3.3.

Figure 4 displays the coverage results obtained using the CRF method and the improvements gained from the inclusion of the length feature and the web search hits. As we can see the CRF gives a coverage 79.9%. Both length model and web search engine show significant improvement over the CRF baseline and the coverage increases to 88.3%.

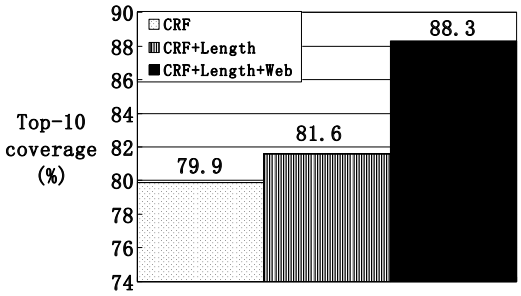


Figure 4: Results of different methods

6 Conclusions and Future work

The CRF works well in generating abbreviations for organization names, while both length model and web search engine further improve the performance.

We are going to perform word clustering or character clustering to alleviate the data sparseness problem. Also we notice that multiple abbreviations for single full-name is very common, such as "中国中央电视台" (China central television) with abbreviations "央视" and "中央台". We plan to collect multiple abbreviations for reference. After that we are going to combine the abbreviation modeling in the voice search system to alleviate the weakness of speech recognition for unknown abbreviation words, which are unlikely to be correctly recognized due to the out of vocabulary problem.

References

Jing-shin Chang and Yu-Tso Lai 2004. *A Preliminary Study on Probabilistic Models for Chinese Abbreviations*. Proceedings of ACL SIGHAN Workshop 2004, pages 9-16.

Guohong Fu, Kang-Kwong Luke, GuoDong Zhou and Ruifeng Xu 2006. *Automatic Expansion of Abbreviations in Chinese News Text*. Lecture Notes in Computer Science, Washington, DC.

John Lafferty, Andrew McCallum, and Fernando Pereira 2001. *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data.*, In Proceedings of International Conference on Machine Learning 2001, pages 282-289

Hiu Wing Doris Lee 2005. *A Study of Automatic Expansion of Chinese Abbreviations*. MA Thesis, The University of Hong Kong.

Zhifei Li and David Yarowsky. 2008. *Unsupervised Translation Induction for Chinese Abbreviations using Monolingual Corpora*. Proceedings of ACL 2008, pages 425-433.

Rila Mandala, Takenobu Tokunaga, and Hozumi Tanaka 2000. *Query expansion using heterogeneous thesauri.*, In Information Processing and Management Volume 36 , Issue 3 2000,Pages 361 - 378

Xu Sun, Houfeng Wang and Yu Zhang 2006. *Chinese Abbreviation-Definition Identification: A SVM Approach Using Context Information*. Lecture Notes in Computer Science,Volume 4182/2006, pages 530-536.

Andrew J. Viterbi 1967. *Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm*. in IEEE Transactions on Information Theory, Volume IT-13, in April, 1967,pages 260-269,

Hanna Wallach 2002. *Efficient Training of Conditional Random Fields*. M. Thesis, University of Edinburgh, 2002.

Hui Yuan and Xianzhong Ruan 2002. *Modern Chinese abbreviation dictionary*. Yuwen press, Beijing, China.

Fast decoding for open vocabulary spoken term detection

[1]B. Ramabhadran,[1]A. Sethy, [2]J. Mamou*[1] B. Kingsbury, [1] U. Chaudhari

[1]IBM T. J. Watson Research Center
Yorktown Heights,NY

[2]IBM Haifa Research Labs
Mount Carmel,Haifa

Abstract

Information retrieval and spoken-term detection from audio such as broadcast news, telephone conversations, conference calls, and meetings are of great interest to the academic, government, and business communities. Motivated by the requirement for high-quality indexes, this study explores the effect of using both word and sub-word information to find in-vocabulary and OOV query terms. It also explores the trade-off between search accuracy and the speed of audio transcription. We present a novel, vocabulary independent, hybrid LVCSR approach to audio indexing and search and show that using phonetic confusions derived from posterior probabilities estimated by a neural network in the retrieval of OOV queries can help in reducing misses. These methods are evaluated on data sets from the 2006 NIST STD task.

1 Introduction

Indexing and retrieval of speech content in various forms such as broadcast news, customer care data and on-line media has gained a lot of interest for a wide range of applications from market intelligence gathering, to customer analytics and online media search. Spoken term detection (STD) is a key information retrieval technology which aims open vocabulary search over large collections of spoken documents. An approach for solving the out-of-vocabulary (OOV) issues (Saraclar and Sproat, 2004) consists of converting speech into phonetic, syllabic or word-fragment transcripts and representing the query as a sequence of phones, syllables or word-fragments respectively. Popular approaches include subword decoding (Clements et al., 2002; Mamou et al., 2007; Seide et al., 2004; Siohan and Bacchiani, 2005) and representations enhanced with phone confusion probabilities and approximate similarity measures (Chaudhari and Picheny, 2007).

*The work done by J. Mamou was partially funded by the EU projects SAPIR and HERMES

2 Fast Decoding Architecture

The first step in converting speech to a searchable index involves the use of an ASR system that produces word, word-fragment or phonetic transcripts. In this paper, the LVCSR system is a discriminatively trained speaker-independent recognizer using PLP-derived features and a quinphone acoustic model with approximately 1200 context dependent states and 30000 Gaussians. The acoustic model is trained on 430 hours of audio from the 1996 and 1997 English Broadcast News Speech corpus (LDC97S44, LDC98S71) and the TDT4 Multilingual Broadcast News Speech corpus (LDC2005S11).

The language model used for decoding is a trigram model with 84087 words trained on a collection of 335M words from the following data sources: Hub4 Language Model data, EARS BN03 closed captions and GALE Broadcast news and conversations data. A word-fragment language model is built on this same data after tokenizing the text to fragments using a fragment inventory of size 21000. A greedy search algorithm assigns the longest possible matching fragment first and iteratively uses the next longest possible fragment until the entire pronunciation of the OOV term has been represented

Proceedings of NAACL HLT 2009: Short Papers, pages 277–280,
Boulder, Colorado, June 2009.

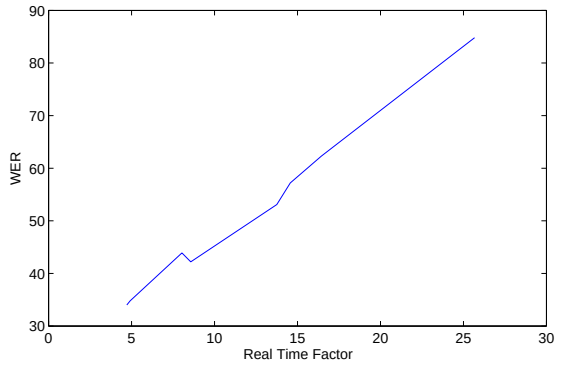


Figure 1: Speed vs WER

by sub-word units.

The speed and accuracy of the decoding are controlled using two forms of pruning. The first is the standard likelihood-based beam pruning that is used in many Viterbi decoders. The second is a form of Gaussian shortlisting in which the Gaussians in the acoustic model are clustered into 1024 clusters, each of which is represented by a single Gaussian. When the decoder gets a new observation vector, it computes the likelihood of the observation under all 1024 cluster models and then ranks the clusters by likelihood. Observation likelihoods are then computed only for those mixture components belonging to the top maxL1 clusters; for components outside this set a default, low likelihood is used. To illustrate the trade-offs in speed vs. accuracy that can be achieved by varying the two pruning parameters, we sweep through different values for the parameters and measure decoding accuracy, reported as word error rate (WER), and decoding speed, reported as times faster than real time (xfRT). For example, a system that operates at 20xfRT will require one minute of time (measured as elapsed time) to process 20 minutes of speech. Figure 1 illustrates this effect on the NIST 2006 Spoken Term Detection Dev06 test set.

3 Lucene Based Indexing and Search

The main difficulty with retrieving information from spoken data is the low accuracy of the transcription, particularly on terms of interest such as named entities and content words. Generally, the accuracy of a transcript is measured by its word error rate (WER), which is characterized by the number of substitutions, deletions, and insertions with respect to the correct audio transcript. Mamou (Mamou et al., 2007) presented the enhancement in recall and precision by searching on word confusion networks instead of considering only the 1-best path word transcript. We used this model for searching in-vocabulary queries.

To handle OOV queries, a combination of word and phonetic search was presented by Mamou (Mamou et al., 2007). In this paper, we explore fuzzy phonetic search extending Lucene[1], an Apache open source search library written in Java, for indexing and search. When searching for these OOVs in word-fragment indexes, they are represented phonetically (and subsequently using word-fragments) using letter-to-phoneme (L2P) rules.

3.1 Indexing

Each transcript is composed of basic units (e.g., word, word-fragment, phones) associated with a begin time, duration and posterior probability. An inverted index is used in a Lucene-based indexing scheme. Each occurrence of a unit of indexing u in a transcript D is indexed on its timestamp. If the posterior probability is provided, we store the confidence level of the occurrence of u at the time t that is evaluated by its posterior probability $Pr(u|t, D)$. Otherwise, we consider its posterior probability to be one. This representation allows the indexing of different types of transcripts into a single index.

3.2 Retrieval

Since the vocabulary of the ASR system used to generate the word transcripts is known, we can easily identify IV and OOV parts of the query. We present two different algorithms, namely, exact and fuzzy search on word-fragment transcripts. For search on word-fragment or phonetic transcripts, the query terms are converted to their word-fragment or phonetic representation.

Candidate lists of each query unit are extracted from the inverted index. For fuzzy search, we retrieve several fuzzy matches from the inverted index for each unit of the query using the edit distance weighted by the substitution costs provided by the confusion matrix. Only the matches whose weighted

[1]http://lucene.apache.org/

edit distance is below a given threshold are returned. We use a dynamic programming algorithm to incorporate the confusion costs specified in the matrix in the distance computation. Our implementation is fail-fast since the procedure is aborted if it is discovered that the minimal cost between the sequences is greater than a certain threshold.

The score of each occurrence aggregates the posterior probability of each indexed unit. The occurrence of each unit is also weighted (user defined weight) according to its type, for example, a higher weight can be assigned to word matches instead of word-fragment or phonetic matches. Given the nature of the index, a match for any query term cannot span across two consecutively indexed units.

3.3 Hybrid WordFragment Indexing

For the hybrid system we limited the word portion of the ASR system's lexicon to the 21K most frequent (frequency greater than 5) words in the acoustic training data. This resulted in roughly 11M (3.1%) OOV tokens in the hybrid LM training set and 1127(2.5%) OOV tokens in the evaluation set. A relative entropy criterion described in (Siohan and Bacchiani, 2005) based on a 5-gram phone language model was used to identify fragments. We selected 21K fragments to complement the 21K words resulting in a composite 42K vocabulary. The language model text (11M (3.1%) fragment tokens and 320M word tokens) was tokenized to contain words and word-fragments (for the OOVs) and the resulting hybrid LM was used in conjunction with the acoustic models described in Section 2.

4 Neural Network Based Posteriors for Fuzzy Search

In assessing the match of decoded transcripts with search queries, recognition errors must be accounted for. One method relies on converting both the decoded transcripts and queries into phonetic representations and modeling the confusion between phones, typically represented as a confusion matrix. In this work, we derive this matrix from broadcast news development data. In particular, two systems: HMM based automatic speech recognition (ASR) (Chaudhari and Picheny, 2007) and a neural network based acoustic model (Kingsbury, 2009), are used to analyze the data and the results are compared to produce confusion estimates.

Let $X = \{x_t\}$ represent the input feature frames and $\mathcal{S}$ the set of context dependent HMM states. Associated with $\mathcal{S}$ is a many to one map $\mathbf{M}$ from each member $s_j \in \mathcal{S}$ to a phone in the phone set $p_k \in \mathcal{P}$. This map collapses the beginning, middle, and end context dependent states to the central phone identity. The ASR system is used to generate a state based alignment of the development data to the training transcripts. This results in a sequence of state labels (classes) $\{\mathbf{s}_t\}$, $\mathbf{s}_t \in \mathcal{S}$, one for each frame of the input data. Note that the aligned states are collapsed to the phone identity with $\mathbf{M}$, so the frame class labels are given by $\{c_t\}$, $c_t \in \mathcal{P}$.

Corresponding to each frame, we also use the state posteriors derived from the output of a Neural Network acoustic model and the prior probabilities computed on the training set. Define $X_t = \{\ldots, x_t, \ldots\}$ to be the sub-sequence of the input speech frames centered around time index t. The neural network takes X_t as input and produces

$$l_t(s_j) = y(s_j|X_t) - l(s_j), s_j \in \mathcal{S}$$

where y is the neural network output and l is the prior probability, both in the log domain. Again, the state labels are mapped using $\mathcal{M}$, so the above posterior is interpreted as that for the collapsed phone:

$$l_t(s_j) \equiv l_t(\mathcal{M}(s_j)) = l_t(p_j), p_j = \mathbf{M}(s_j).$$

The result of both analyses gives the following set of associations:

$$\begin{aligned} c_0 &\leftrightarrow l_0(p_0), l_0(p_1), l_0(p_2), \ldots \\ c_1 &\leftrightarrow l_1(p_0), l_1(p_1), l_1(p_2), \ldots \\ &\vdots \\ c_t &\leftrightarrow l_t(p_0), l_t(p_1), l_t(p_2), \ldots \end{aligned}$$

Each log posterior $l_i(p_j)$ is converted into a count

$$n_{i,j} = ceil[N \times e^{l_i(p_j)}],$$

where N is a large constant, i ranges over the time index, and j ranges over the context dependent states. From the counts, the confusion matrix entries are computed. The total count for each state is

$$n_j(k) = \sum_{i:c_i=p_j} n_{i,k},$$

where k is an index over the states.

$$\begin{bmatrix} n_1(1) & n_1(2) & \dots \\ n_2(1) & n_2(2) & \dots \\ & \cdot & \\ & \cdot & \end{bmatrix}$$

The rows of the above matrix correspond to the reference and the columns to the observations. By normalizing the rows, the entries can be interpreted as "probability" of an observed phone (indicated by the column) given the true phone.

5 Experiments and Results

The performance of a spoken term detection system is measured using DET curves that plot the trade-off between false alarms (FAs) and misses. This NIST STD 2006 evaluation metric used Actual/Maximum Term Weighted Value (ATWV/MTWV) that allows one to weight FAs and Misses per the needs of the task at hand (NIST, 2006).

Figure 2 illustrates the effect of speed on ATWV on the NIST STD 2006 Dev06 data set using 1107 query terms. As the speed of indexing is increased to many times faster than real time, the WER increases, which in turn decreases the ATWV measure. It can be seen that the use of word-fragments improves the performance on OOV queries thus making the combined search better than simple word search. The primary advantage of using a hybrid decoding scheme over a separate word and fragment based decoding scheme is the speed of transforming the audio into indexable units. The blue line in the figure illustrates that when using a hybrid setup, the same performance can be achieved at speeds twice as fast. For example, with the combined search on two different decodes, an ATWV of 0.1 can be achieved when indexing at a speed 15 times faster than real time, but with a hybrid system, the same performance can be reached at an indexing speed 30 times faster than real time. The ATWV on the hybrid system also degrades gracefully with faster speeds when compared to separate word and word-fragment systems. Preliminary results indicate that fuzzy search on one best output gives the same ATWV performance as exact search (Figure 2) on consensus output. Also, a closer look at the retrieval results of OOV terms revealed that many more OOVs are retrieved with the fuzzy search.

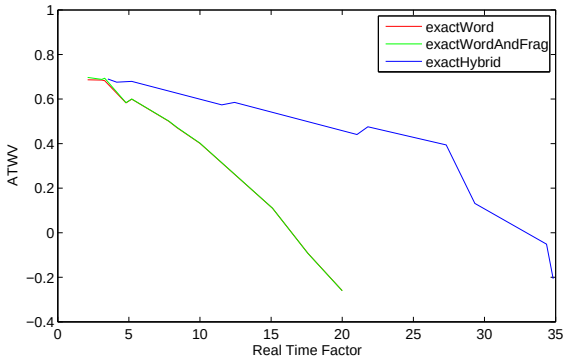


Figure 2: Effect of WER on ATWV. Note that the cuves for exactWord and exactWordAndFrag lie on top of each other.

6 CONCLUSION

In this paper, we have presented the effect of rapid decoding on a spoken term detection task. We have demonstrated that hybrid systems perform well and fuzzy search with phone confusion probabilities help in OOV retrieval.

References

U. V. Chaudhari and M. Picheny. 2007. Improvements in phone based audio search via constrained match with high order confusion estimates. In *Proc. of ASRU*.

M. Clements, S. Robertson, and M. S. Miller. 2002. Phonetic searching applied to on-line distance learning modules. In *Proc. of IEEE Digital Signal Processing Workshop*.

B. Kingsbury. 2009. Lattice-based optimization of sequence classification criteria for neural-network acoustic modeling. In *Proc. of ICASSP*.

J. Mamou, B. Ramabhadran, and O. Siohan. 2007. Vocabulary independent spoken term detection. In *Proc. of ACM SIGIR*.

NIST. 2006. The spoken term detection (STD) 2006 evaluation plan. http://www.nist.gov/speech/tests/std/docs/std06-evalplan-v10.pdf.

M. Saraclar and R. Sproat. 2004. Lattice-based search for spoken utterance retrieval. In *Proc. HLT-NAACL*.

F. Seide, P. Yu, C. Ma, and E. Chang. 2004. Vocabulary-independent search in spontaneous speech. In *Proc. of ICASSP*.

O. Siohan and M. Bacchiani. 2005. Fast vocabulary independent audio search using path based graph indexing. In *Proc. of Interspeech*.

Tightly coupling Speech Recognition and Search

Taniya Mishra
AT&T Labs-Research
180 Park Ave
Florham Park, NJ 07932
taniya@research.att.com

Srinivas Bangalore
AT&T Labs-Research
180 Park Ave
Florham Park, NJ 07932
srini@research.att.com

Abstract

In this paper, we discuss the benefits of tightly coupling speech recognition and search components in the context of a speech-driven search application. We demonstrate that by incorporating constraints from the information repository that is being searched not only improves the speech recognition accuracy but also results in higher search accuracy.

1 Introduction

With the exponential growth in the use of mobile devices in recent years, the need for speech-driven interfaces is becoming apparent. The limited screen space and soft keyboards of mobile devices make it cumbersome to type in text input. Furthermore, by the *mobile* nature of these devices, users often would like to use them in hands-busy environments, ruling out the possibility of typing text.

In this paper, we focus on the problem of speech-driven search to access information repositories using mobile devices. Such an application typically uses a speech recognizer (ASR) for transforming the user's speech input to text and a search component that uses the resulting text as a query to retrieve the relevant documents from the information repository. For the purposes of this paper, we use the business listings containing the name, address and phone number of businesses as the information repository.

Most of the literature on speech-driven search applications that are available in the consumer market (Acero et al., 2008; Bacchiani et al., 2008; VLingo FIND, 2009) have quite rightly emphasized the importance of the robustness of the ASR language model and the data needed to build such a robust language model. We acknowledge that this is a significant issue for building such systems, and we provide our approach to creating a language model. However, in contrast to most of these systems that treat speech-driven search to be largely an ASR problem followed by a Search problem, in this paper, we show the benefits of tightly coupling ASR and Search tasks and illustrate techniques to improve the accuracy of both components by exploiting the co-constraints between the two components.

The outline of the paper is as follows. In Section 2, we discuss the set up of our speech-driven application. In Section 3, we discuss our method to integrating the speech and search components. We present the results of the experiments in Section 4 and conclude in Section 5.

2 Speech-driven Search

We describe the speech-driven search application in this section. The user of this application provides a speech utterance to a mobile device intending to search for the address and phone number of a business. The speech utterance typically contains a business name, optionally followed by a city and state to indicate the location of the business (e.g. *pizza hut near urbana illinois*.). User input with a business category (*laundromats in madison*) and without location information (*hospitals*) are some variants supported by this application. The result of ASR is used to search a business listing database of over 10 million entries to retrieve the entries pertinent to the user query.

The ASR used to recognize these utterances incorporates an acoustic model adapted to speech collected from mobile devices and a trigram language model that is built from over 10 million text query logs obtained from the web-based text-driven version of this application. The 1-best speech recognition output is used to retrieve the relevant business listing entries.

Proceedings of NAACL HLT 2009: Short Papers, pages 281–284,
Boulder, Colorado, June 2009.

3 Tightly coupling ASR and Search

As mentioned earlier, most of the speech-driven search systems use the the 1-best output from the ASR as the query for the search component. Given that ASR 1-best output is likely to be erroneous, this serialization of the ASR and search components might result in sub-optimal search accuracy. As will be shown in our experiments, the oracle word/phrase accuracy using n-best hypotheses is far greater than the 1-best output. However, using each of the n-best hypothesis as a query to the search component is computationally sub-optimal since the strings in the n-best hypotheses usually share large subsequences with each other. A lattice representation of the ASR output, in particular, a word-confusion network (WCN) transformation of the lattice, compactly encodes the n-best hypothesis with the flexibility of pruning alternatives at each word position. An example of a WCN is shown in Figure 1. In order to obtain a measure of the ambiguity per word position in the WCN, we define the (average) *arc density* of a WCN as the ratio of the total number of arcs to the number of states in the WCN. As can be seen, with very small increase in arc density, the number of paths that are encoded in the WCN can be increased exponentially. In Figure 2, we show the improvement in oracle-path word and phrase accuracies as a function of the arc density for our data set. Oracle-path is a path in the WCN that has the least edit-distance (Levenshtein, 1966) to the reference string. It is interesting to note that the oracle accuracies can be improved by almost 10% absolute over the 1-best accuracy with small increase in the arc density.

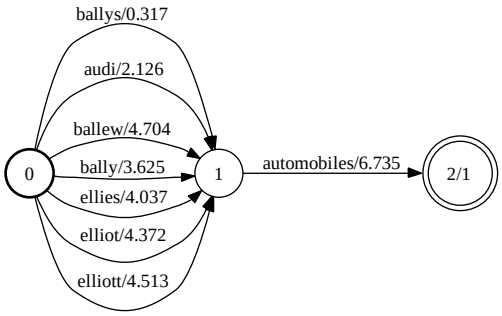


Figure 1: A sample word confusion network

3.1 Representing Search Index as an FST

In order to exploit WCNs for Search, we have implemented our own search engine instead of using an off-the-shelf search engine such as Lucene (Hatcher and Gospodnetic., 2004). We index each business listing (d) in our data that we intend to search using the words (w_d) in that listing. The pair (w_d, d) is assigned a weight ($c_{(w_d,d)}$) using different metrics, including the standard $tf * idf$, as explained below. This index is represented as a weighted finite-state transducer (*SearchFST*) as shown in Figure 3 where w_d is the input symbol, d is the output symbol and $c_{(w_d,d)}$ is the weight of that arc.

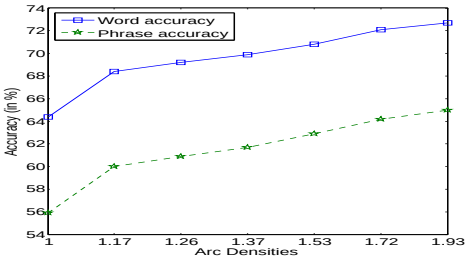


Figure 2: Oracle accuracy graph for the WCNs at different arc densities

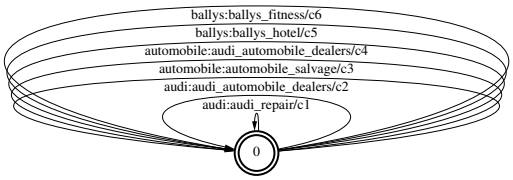


Figure 3: An example of an FST representing the search index

3.2 Relevance Metrics

In this section, we describe six different weighting metrics used to determine the relevance of a document for a given query word that we have experimented with in this paper.

idf_w: idf_w refers to the inverse document frequency of the word, w, which is computed as $ln(D/d_w)$, where D refers to the total number of documents in the collection, and d_w refers to the total number of documents in the collection that contain the word, w (Robertson and Jones, 1997; Robertson, 2004).

atf_w: atf_w refers to average term frequency, which is computed as cf_w/d_w (Pirkola et al., 2002).

$cf_w \times idf_w$: Here cf_w refers to the collection frequency, which is simply the total number of occurrences of the word, w in the collection.

$atf_w \times idf_w$**:** (Each term as described above).

$\sum \frac{f_{w,d}}{|d_w|} \times idf_w$**:** Here $f_{w,d}$ refers to the frequency of the word, w, in the document, d, whereas $|d_w|$ is the length of the document, d, in which the word, w, occurs.

$\frac{cf_w}{\sum |d_w|} \times idf_w$**:** (Each term as described above).

3.3 Search

By composing a query (*Qfst*) (either a 1-best string represented as a finite-state acceptor, or a WCN), with the *SearchFST*, we obtain all the arcs $(w_q, d_{w_q}, c_{(w_q,d_{w_q})})$ where w_q is a query word, d_{w_q} is a listing with the query word and, $c_{(w_q,d_{w_q})}$ is the weight associated with that pair. Using this information, we aggregate the weight for a listing (d_q) across all query words and rank the retrieved listings in the descending order of this aggregated weight. We select the top N listings from this ranked list. The query composition, listing weight aggregation and selection of top N listings are computed with finite-state transducer operations.

In Figure 4, we illustrate the result of reranking the WCN shown in Figure 1 using the search relevance weights of each word in the WCN. It must be noted that the least cost path[1] for the WCN in Figure 1 is *ballys automobiles* while the reranked 1-best output in Figure 4 is *audi automobiles*. Given that the user voice query was *audi automobiles*, the listings retrieved from the 1-best output after reranking are much more relevant than those retrieved before reranking, as shown in Table 1.

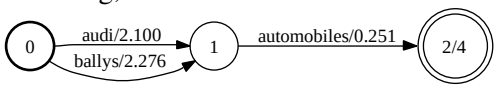


Figure 4: A WCN rescored using word-level search relevance weights.

4 Experiments and Results

We took 852 speech queries collected from users using a mobile device based speech search application. We ran the speech recognizer on these queries using the language model described in Section 2 and created word-confusion networks such as those illustrated in Figure 1. These 852 utterances were divided into 300 utterances for the development set and 552 for the test set.

[1]We transform the scores into costs and search for minimum cost paths.

Before rescoring	After rescoring
ballys intl los angeles ca	auburn audi repair auburn wa
ballys las vegas las vegas nv	audi bellevue repair bellevue wa
ballys las health spa las vegas nv	university audi seattle wa
ballys cleaners palm desert ca	beverly hills audi los angeles ca
ballys brothers yorba linda ca	audi independent repairs by eurotech livermore ca

Table 1: Listings retrieved for query *audi automobiles* before and after ASR WCNs were rescored using search relevance weights.

4.1 ASR Experiments

The baseline ASR word and sentence (complete string) accuracies on the development set are 63.1% and 57.0% while those on the test set are 65.1% and 55.3% respectively.

Metric	Word Acc.	Sent. Acc.	Scaling Factor	AD
idf_w	63.1	57.0	10^-3	all
$cf_w \times idf_w$	63.5	58.3	$15 * 10^-4$	1.37
atf_w	63.6	57.3	1	all
$atf_w \times idf$	63.1	57.0	10^-3	all
$\sum \frac{f_{w,d}}{\|df_w\|} \times idf$	63.9	58.3	$15 * 10^-4$	1.25
$\frac{cf_w}{\sum \|df_w\|} \times idf_w$	63.5	57.3	1	all

Table 2: Performance of the metrics used for rescoring the WCNs output by ASR. (AD refers to *arc density*.)

In Table 2, we summarize the improvements obtained by rescoring the ASR WCNs based on the different metrics used for computing the word scores according to the search criteria. The largest improvement in word and sentence accuracies is obtained by using the rescoring metric: $\sum \frac{f_{w,d}}{|df_w|} \times idf$. The word-level accuracy improved from the baseline accuracy of 63.1% to 63.9% after rescoring while the sentence-level accuracy improved from 57.0% to 58.3%. Thus, this rescoring metric, and the corresponding pruning AD and the scaling factor was used to rerank the 552 WCNs in the test set. After rescoring, on the test set, the word-level accuracy improved from 65.1% to 65.9% and sentence-level accuracy improved from 55.3% to 56.2%.

Number of documents	Scores	Baseline	Reranked
All Documents	Precision	0.708	0.728
	Recall	0.728	0.742
	F-Score	0.718	0.735

Table 3: Table showing the relevancy of the search results obtained by the baseline ASR output compared to those obtained by the reranked ASR output.

4.2 Search Experiments

To analyze the Search accuracy of the baseline ASR output in comparison to the ASR output, reranked using the $\sum \frac{f_{w,d}}{|df_w|} \times idf$ reranking metric, we used each of the two sets of ASR outputs (i.e., baseline and reranked) as queries to our search engine, *SearchFST* (described in Section 3). For the search results produced by each set of queries, we computed the precision, recall, and F-score values of the listings retrieved with respect to the listings retrieved by the set of human transcribed queries (*Reference*). The precision, recall, and F-scores for the baseline ASR output and the reranked ASR output, averaged across each set, is presented in Table 3. For the purposes of this experiment, we assume that the set returned by our *SearchFST* for the human transcribed set of queries is the reference search set. This is however an approximation for a human annotated search set.

In Table 3, by comparing the search accuracy scores corresponding to the baseline ASR output to those corresponding to the reranked ASR output, we see that reranking the ASR output using the information repository produces a substantial improvement in the accuracy of the search results.

It is interesting to note that even though the reranking of the ASR as shown in Table 2 is of the order of 1%, the improvement in Search accuracy is substantially higher. This indicates to the fact that exploiting constraints from both components results in improving the recognition accuracy of that subset of words that are more relevant for Search.

5 Conclusion

In this paper, we have presented techniques for tightly coupling ASR and Search. The central idea behind these techniques is to rerank the ASR output using the constraints (encoded as relevance metrics) from the Search task. The relevance metric that best improved accuracy is $\sum \frac{f_{w,d}}{|d_w|} \times idf_w$, as determined on our development set. Using this metric to rerank the ASR output of our test set, we improved ASR accuracy from 65.1% to 65.9% at the word-level and from 55.3% to 56.2% at the phrase level. This reranking also improved the F-score of the search component from 0.718 to 0.735. These results bear out our expectation that tightly coupling ASR and Search can improve the accuracy of both components.

Encouraged by the results of our experiments, we plan to explore other relevance metrics that can encode more sophisticated constraints such as the relative coherence of the terms within a query.

Acknowledgments

The data used in this work is partly derived from the Speak4It voice search prototype. We wish to thank every member of that team for having deployed that voice search system.

References

A. Acero, N. Bernstein, R.Chambers, Y. Ju, X. Li, J. Odell, O. Scholtz P. Nguyen, and G. Zweig. 2008. Live search for mobile: Web services by voice on the cellphone. In *Proceedings of ICASSP 2008*, Las Vegas.

M. Bacchiani, F. Beaufays, J. Schalkwyk, M. Schuster, and B. Strope. 2008. Deploying GOOG-411: Early lesstons in data, measurement and testing. In *Proceedings of ICASSP 2008*, Las Vegas.

E. Hatcher and O. Gospodnetic. 2004. *Lucene in Action (In Action series)*. Manning Publications Co., Greenwich, CT, USA.

V.I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertion and reversals. *Soviet Physics Doklady*, 10:707–710.

A. Pirkola, E. Lepaänen, and K. Järvelin. 2002. The "ratf" formula (kwok's formula): exploiting average term frequency in cross-language retrieval. *Information Research*, 7(2).

S. E. Robertson and K. Sparck Jones. 1997. Simple proven approaches to text retrieval. Technical report, Cambridge University.

Stephen Robertson. 2004. Understanding inverse document frequency: On theoretical arguments for idf. *Journal of Documentation*, 60.

VLingo FIND, 2009. *http://www.vlingomobile.com/downloads.html*.

Author Index